MODELING SEIZURES WITH A CONDUCTANCE-BASED INTEGRATE-AND-FIRE NEURONAL NETWORK MODEL

by

REBECCA J GAFF
(Under the direction of Andrew Sornborger)

ABSTRACT

Imaging research of seizures in Zebrafish brains involves tracing calcium levels as a by-product of neuronal firing using genetically encoded fluorescent calcium indicators. The imaging data suggests the seizures move in a wave-like pattern through the various parts of the brain. Using this data, researchers can predict the frequency of firing rates within the brain.

The purpose of this thesis is to create a mathematical model that demonstrates a similar wave-like pattern at similar firing rate frequencies using a neuronal network of integrate-and-fire neurons. Calcium and its emission ratio are also modeled to connect the simulated data with the experimental data.

INDEX WORDS:
Seizures; Neuronal Networks; Integrate-and-fire neurons; mathematical modeling; Zebrafish brain imaging; Calcium levels and emission ratios

MODELING SEIZURES WITH A CONDUCTANCE-BASED INTEGRATE-AND-FIRE NEURONAL NETWORK MODEL

by

REBECCA J. GAFF

B.S., University of Florida, 2006

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTERS OF ARTS

ATHENS, GEORIGA

2010

MODELING SEIZURES WITH A CONDUCTANCE-BASED INTEGRATE-AND-FIRE NEURONAL NETWORK MODEL

by

REBECCA J GAFF

Approved:

Major Professor: Andrew Sornborger

Committee: Malcolm Adams
Caner Kazanci

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
University of Georgia
July 2010

Table of Contents

List of Figures

Page

CHAPTER 1

INTRODUCTION AND LITERATURE REVIEW

Seizures have been studied for decades and yet there is still a lot of mystery surrounding them. We know seizures in the brain are caused by too much excitation in the electrical activity of its neurons. Beginning with Hodgkin and Huxley, we have mathematical models that represent neurons as an electrical circuit giving insight to its electrical activity. Using these models, we can calculate a neuron's voltage and conductance over time. We can also model populations of neurons and see how these connections can cause changes in voltage and conductance of surrounding neurons.

Looking at populations of neurons, there are still some questions to be answered. We want to know how neurons communicate with one another during a seizure. And since there are different ways to model the connections between and within populations of neurons, what is the effect of the connectivity structure on how seizures move through the brain? Recent developments in brain imaging techniques have allowed researchers to observe the propagation of seizures by imaging calcium released during neuronal firing.

The motivation for my thesis stems from research on seizure imaging in the larval Zebrafish brain conducted by Andrew Sornborger and James Lauderdale at the University of Georgia. Using transgenic Zebrafish with a genetically encoded calcium indicator, Sornborger and Lauderdale are able to visualize the movement of a seizure

by imaging the released calcium, a by-product of neuronal firing. The results suggest that the seizure propagates as a wave through different parts of the brain. In the research presented here, I attempt to use proven models of neuronal networks and recently developed techniques to investigate seizure propagation in a neuronal network.

In this chapter I will present background literature on the properties of neurons and how they are modeled. I will describe the structure of a neuron and present three sets of equations of varying complexity that model the electrical activity of a neuron in varying levels of detail.

1.1 The Neuron

The neuron is the elementary processing unit in the brain. Its main function is to transmit and receive information. The structure of the neuron with its axon, synapse, and dendrite components give it this ability [1].

The purpose of the axon is to transmit electrical activity from the cell body to the synapses. The axon is connected to the soma at the axon hillock. The axon hillock is where neuronal signals begin. Neuronal signals consist of short electrical pulses, also known as action potentials or spikes, which propagate along the axon. The number and timing of these spikes is what is most important since the action potentials themselves to do not retain any information. The axon's long, tube-like structure, often with a myelin coating, allows it to quickly transmit the action potentials to its pre-synaptic terminal that is located at the distal end of the axon [1,2,3].

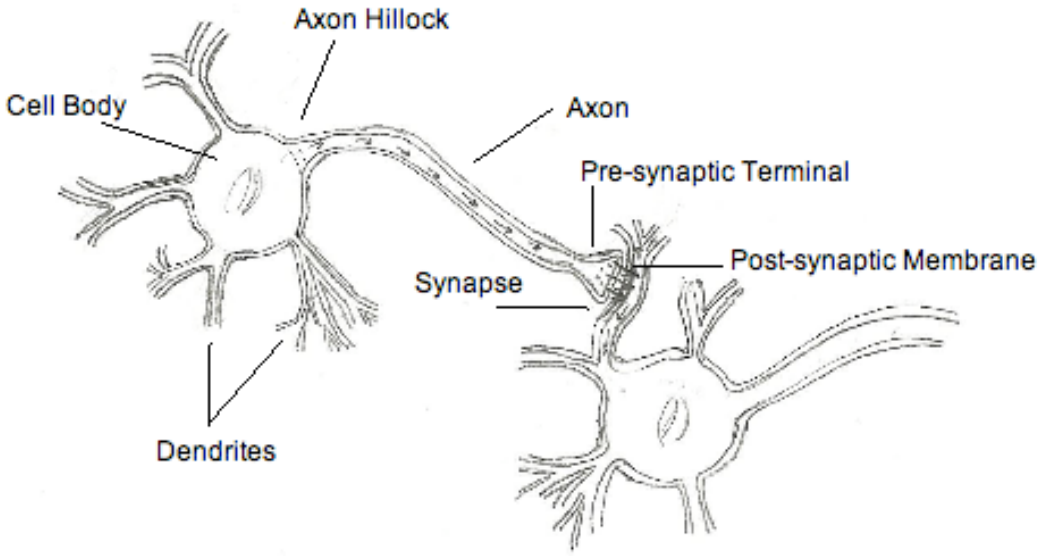


FIGURE 1.1 STRUCTURE OF A NEURON. The tube-like structure of the axon is built to easily receive and transfer information. The synapse is where information is transferred from the pre-synaptic neuron to the post-synaptic neuron [1,3].

The synapse is the junction where communication between the pre-synaptic neuron and the post-synaptic neuron is transmitted. When a series of spikes arrive at the synapse, it causes neurotransmitters to be released across the synaptic cleft (the space between the pre- and post- synaptic neuron). Once the neurotransmitters have reached the post-synaptic neuron, they attach themselves to receptor sites opening ion channels. This ion influx inevitably leads to changes in the membrane potential of the post-synaptic neuron [1,2,3].

Dendrites are thin branch-like projections stemming from the cell body and are mainly responsible for receiving signals from a spiking neuron. Dendrites contain special “receptor sites” which allow the neuron to receive information from other neurons. The information in the form of post-synaptic conductances then gets transmitted to the cell body where the conductances are combined. Once the combined

conductances reaches a threshold, the cell sends out an action potential (spike) through the axon to once again propagate the information to other neurons [1,2,3].

1.2 The Hodgkin-Huxley Model

In the 1950s, Alan Hodgkin and Andrew Huxley developed the first mathematical model for measuring the electrical potential of the neuronal cell membrane in an axon. Their model was initially designed to predict the dynamics of action potentials in the giant axon of a squid. Since their work and ideas are so commonly used among other excitable cell models, it is considered the foundational model of computational neuroscience [4,5]. In 1963, Hodgkin and Huxley were recognized for their contribution as recipients of the Nobel Prize in Medicine [6].

From the Hodgkin-Huxley equations, simpler models such as the FitzHugh-Nagumo and Integrate-and-Fire models have been developed that are more conducive to mathematical analysis [4,5]. In this section we will introduce and give a brief overview of the Hodgkin-Huxley equations.

A cell membrane can be modeled as a capacitor in parallel[1] with an ionic current

$$C_m \frac{dv}{dt} = -I_{ion}(v,t) + I_{app} \tag{1.2.1}$$

[1] Circuits are connected in parallel if voltage drops across them are the same.

where C_m is membrane capacitance, $\frac{dv}{dt}$ is the rate of change of voltage[1] across the capacitor, I_{ion} is the ionic current due to active ion channels in the cell membrane, and I_{app} is an applied current that can drive the voltage. Hodgkin and Huxley described two main ionic currents in their model: sodium, I_{Na}, and potassium, I_K. They lumped the others together into one term, the leakage current, I_L, because of the minor impact they had on the system. Using Ohm's[2] Law and the fact that conductance, g_{ion}, is the inverse of resistance, we can write (1.2.1) as

$$C_m \frac{dv}{dt} = -g_{Na}(v - V_{Na}) - g_k(v - V_k) - g_L(v - V_L) + I_{app} \qquad (1.2.2)$$

Results from experiments with the squid giant axon did not quite follow this model for stronger applied currents. Therefore, Hodgkin and Huxley determined that the conductance of the potassium and sodium ions were not constant but instead were dependent on voltage [5].

The development of the voltage clamp was important to further investigation. The voltage clamp had the ability to fix membrane potential, and then measure the amount of current needed to keep the voltage constant. This allowed Hodgkin and Huxley to eliminate voltage changes while measuring ionic conductance changes. Hence, the conductance was only dependent on time. After some assumptions about the nature of the sodium and potassium currents, they were able to measure the total ionic currents

[1] Note: v represents membrane potential, i.e. the difference between internal and external cell voltage.
[2] Ohm's Law states the relationship between voltage, resistance, and current is $v = IR$.

and the ratios of each current[1]. From these currents, and the known voltages, they obtained conductance values for sodium and potassium. Based on experimental data, they were able to define equations to replicate potassium and sodium conductance changes [5].

Potassium conductance behavior follows a sigmoidal increase during increased voltage and then an exponential fall during a decrease. Hodgkin and Huxley proposed that the potassium conductance takes on the form

$$g_K = \overline{g}_K n^4$$

where $\overline{g}_K$ is some constant and n, known as the potassium activation, obeys the differential equation,

$$\tau_n(v)\frac{dn}{dt} = n_\infty(v) - n$$

also written as,

$$\frac{dn}{dt} = \alpha_n(v)(1-n) - \beta_n(v)n \qquad (1.2.3)$$

for

$$\tau_n(v) = \frac{1}{\alpha_n(v) + \beta_n(v)} \ ,$$

$$n_\infty(v) = \frac{\alpha_n(v)}{\alpha_n(v) + \beta_n(v)}$$

[1] See [5] for a more detailed explanation of assumptions and conclusions.

and $v = V - V_{rest}$, the difference between the membrane potential and the resting potential. The functions n_∞ and τ_n are determined from experimental data [5].

Sodium conductance follows a different pattern from potassium. During the same time step of increased constant voltage, g_{Na} increases at the beginning and then begins to decrease. Therefore, Hodgkin and Huxley proposed that the sodium conductance takes the form

$$g_{Na}(v) = \overline{g}_{Na} m^3 h$$

where the time-dependent behavior of sodium activation, m, and sodium inactivation, h, obey the differential equations

$$\frac{dm}{dt} = \alpha_m (1 - m) - \beta_m m \qquad (1.2.4)$$

$$\frac{dh}{dt} = \alpha_h (1 - h) - \beta_h h \ . \qquad (1.2.5)$$

For any fixed voltage step, α_m (α_h) and β_m (β_h) are found by fitting a function to experimental data [5].

The collection of differential equations (1.2.2) – (1.2.5) is a four dimensional system. The dynamics of this system depend on the variables *m*, *n*, and *h* and the difference in time constants, τ_m, τ_n, and τ_h. The time constant τ_m is smaller than the other time constants and it causes $m(t)$ to respond more quickly to changes in potential. For example, if the potential v is raised only slightly, the system will return to its equilibrium. However, with a strong enough stimulating current, the potential will rise

above threshold and before returning to rest will have an increase in sodium activation, m. To summarize the influences of m, n, and h, Figure 1.2 shows the action potential following a superthreshold stimulus (a stimulus that causes a spike) and Figure 1.3 demonstrates the changes in $m(t)$, $n(t)$ and $h(t)$ during the same action potential [5].

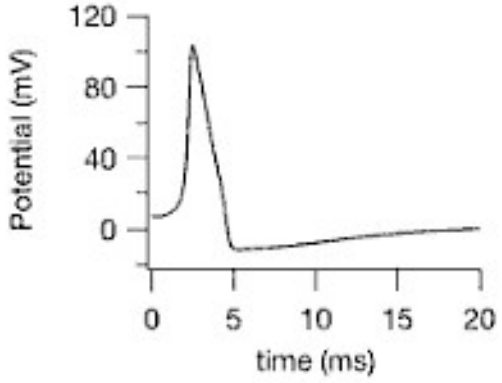


FIGURE 1.2
Action potential of the Hodgkin-Huxley equations during a superthreshold stimulus [5].

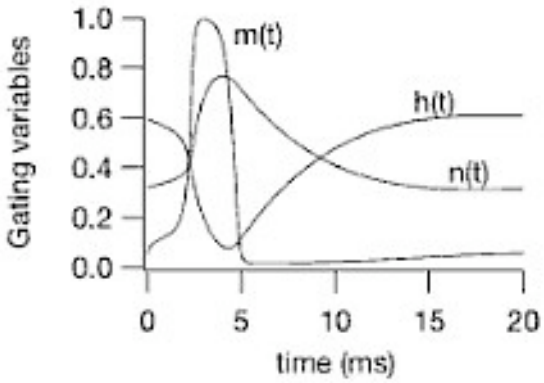


FIGURE 1.3
Plot of the gating variables during the same action potential from Fig 1.2 [5].

A four dimensional system is very difficult to analyze quantitatively. FitzHugh among others, was able to reduce the dimension of the system and still preserve the qualitative features of the Hodgkin-Huxley equations. In the next section we will take a closer look at the development of the FitzHugh-Nagumo Model.

1.3 FITZHUGH-NAGUMO MODEL

FitzHugh attempted to reduce the dimensions of the Hodgkin-Huxley equations while maintaining the qualitative behavior of the fast-slow phase-plane. Since both the

membrane potential, v, and sodium activation, m, activate quickly, while the sodium inactivation, h, and potassium activation, n, act slowly, his strategy was to combine the fast and slow variables together. Hence, the model reduces to two variables, one fast, v, and one slow, w. The fast variable, which causes excitation, has a cubic nullcline given by a cubic function, $f(v,w)$. The slow variable, known as the recovery variable, has a strictly increasing nullcline given by a linear function, $g(v,w)$. Figure 1.4 shows a schematic diagram of the phase plane [5].

The FitzHugh-Nagumo model comes from the model of the cell membrane as a circuit. Nagumo built a circuit consisting of a capacitor, a current-voltage device to represent the behavior of v, and a resistor, inductor, and battery combination to symbolize the behavior of w. He used a tunnel diode[1], which is a

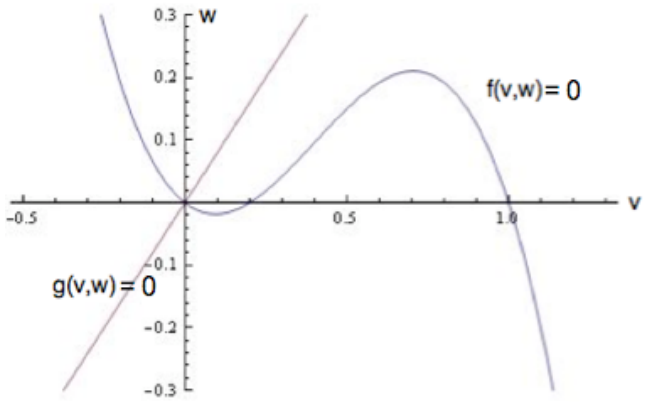


FIGURE 1.4. A schematic diagram of the general phase plane for the FitzHugh-Nagumo model.

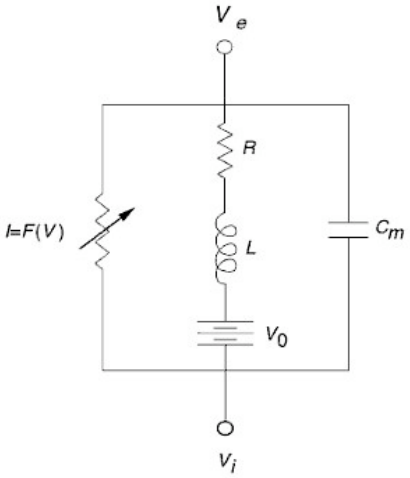


FIGURE 1.5 Diagram of the circuit modeled by Nagumo to represent the FitzHugh-Nagumo equations [5].

[1] See [7] pg. 417-420 for more information on tunnel diodes.

type of semi-conductor diode capable of very fast operation, to characterize the nonlinear element [5,7]. Figure 1.5 shows the diagram of his circuit. Kirchhoff's Laws, which characterize conservation of charge and energy for circuits [8], can be used to write equations for the cell membrane circuit diagram given in Figure 1.5:

$$C_m \frac{dV}{d\tau} + F(V) + i = -I_{app} \tag{1.3.1}$$

$$L\frac{di}{d\tau} + Ri = V - V_0 \tag{1.3.2}$$

where i is the resistor-inductor current, $V = V_i - V_e$ is the membrane potential, I_{app} is the applied external current, and V_0 is the potential gain in voltage across the battery. The function $F(V)$ is cubic and has stable solutions to the differential equation, $\frac{dV}{d\tau} = -F(V)$, at it's largest $(V = V_1)$ and smallest $(V = V_0)$ zeros. Letting R_1 be the passive resistance of $F(V)$, $R_1 = \frac{1}{F'(0)}$, and non-dimensionalizing the equations (1.3.1) and (1.3.2) with $v = \frac{V}{V_1}, w = \frac{R_1 i}{V_1}, f(v) = \frac{-R_1 F(V_1 v)}{V_1}$, and $t = \frac{L\tau}{R_1}$, we get the FitzHugh-Nagumo equations

$$\frac{dv}{dt} = \frac{1}{\varepsilon}(f(v) - w - w_0) \tag{1.3.3}$$

$$\frac{dw}{dt} = v - \gamma w - v_0 \tag{1.3.4}$$

where $\varepsilon = \frac{R_1^2 C_m}{L}, w_0 = \frac{R_1 I_0}{V_1}, v_0 = \frac{V_0}{V_1}$, $\gamma = \frac{R}{R_1}$, and I_0 is the applied external current [5]. Since the only condition on $f(v)$ is that it be of cubic form, the easiest choice to pick is a cubic polynomial

$$f(v) = v(v-a)(b-v) \qquad \text{with } |a| < 1 \text{ and } b > 0 \qquad (1.3.5)$$

Now that we have our FitzHugh-Nagumo model with equations (1.3.3) and (1.3.4) we can use phase plane analysis to characterize the different possible phase portraits. The fixed points occur when $\frac{dv}{dt} = 0$ and $\frac{dw}{dt} = 0$. Therefore, $w^* = v^*(v^* - a)(b - v^*)$ and $v^* = \gamma w^*$. Assuming $f(v,w)$ and $g(v,w)$ has exactly one point of intersection, we can assume without loss of generality that $w_0 = 0$ and v_0 = 0 since this is just a shift in coordinates. We can therefore conclude that the only fixed point is $v^* = w^* = 0$. Plotting the two distinctive phase portraits, we can see that either the system is excitable and converges to a stable rest point (Fig 1.6) or a Hopf Bifurcation occurs (Figure 1.8) when the fixed point lies on the middle branch of the cubic nullcline, where it is unstable.

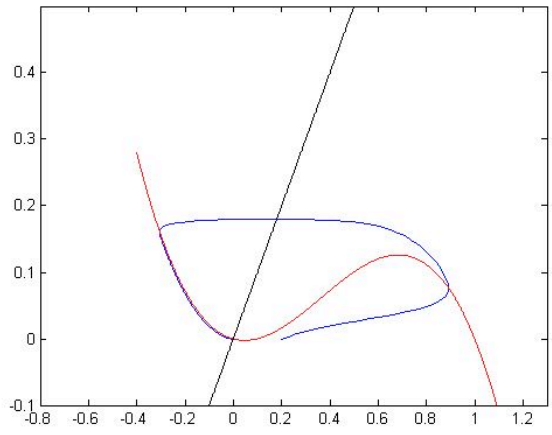


FIGURE 1.6. Stable phase portrait for w_0=v_0=0, γ = .5, a=.1, and b = 1.

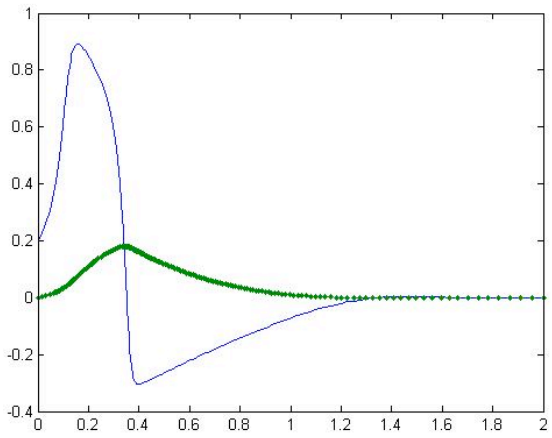


FIGURE 1.7. Solutions to FitzHugh-Nagumo using parameters from Fig 1.6.

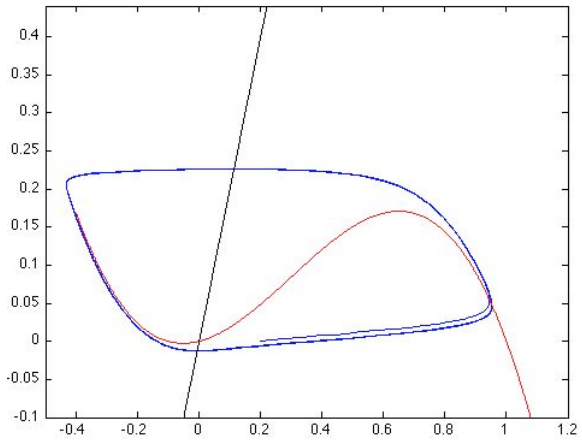


FIGURE 1.8 Unstable phase portrait for $w_0=v_0=0$, γ = .5 , a=-.1, and b=1.

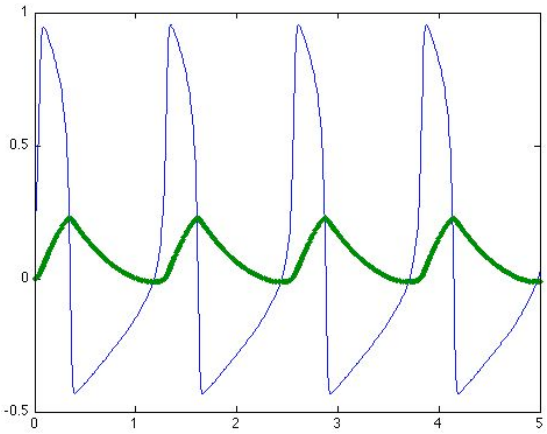


FIGURE 1.9 Solutions to FitzHugh-Nagumo using parameters from Fig 1.8.

The FitzHugh-Nagumo equations were able to reduce the dimension of the Hodgkin-Huxley equations to a two dimensional system. We've seen that with a two-dimensional system analysis is relatively simple. However, the integrate-and-fire model is a further simplification that further simplifies the fast-slow time scale in the membrane potential.

1.4 INTEGRATE-AND-FIRE MODEL

The integrate-and-fire model is a simple spiking cell model that was first investigated by Louis Lapicque. His ideas were published in an article in 1907 [9]. Integrate-and-fire has become a standard model for neuronal network dynamics in large-scale simulations due to its simplicity. It still captures the essence of the more complicated models by retaining an integrating threshold and the generation of spikes after the membrane potential crosses the given threshold.

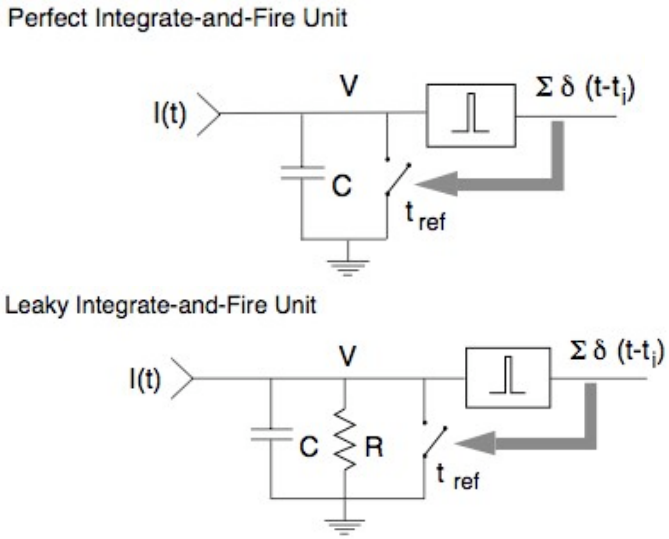


FIGURE 1.10 THE PERFECT AND LEAKY INTEGRATE-AND-FIRE. When $V(t_i) = V_{th}$ an action potential is generated and modeled by the delta function, $\delta(t-t_i)$. [9]

Several types of integrate-and-fire models exist. Two of these models are the perfect and leaky integrate-and-fire (IF) models (see Figure 1.10). The perfect IF model consists of a single capacitance for integrating the input current. Once the membrane potential reaches threshold, a spike, in the form of a delta function, occurs. Then the voltage is reset to rest before it begins integrating again. A refractory period, t_{ref}, can also be added into the model after a spike. The perfect IF neuron can be modeled using the following first order differential equation:

$$C\frac{dv}{dt} = I(t) \tag{1.4.1}$$

where C is capacitance, v is the membrane potential, and I(t) is the input current [9]. Once the membrane potential reaches threshold, the membrane potential is reset and once again follows the dynamics given by Equation 1.4.1. One shortcoming to the model is that it has no account for a leak. If the model receives subthreshold inputs, it

will maintain the voltage level produced by these inputs with no resistance. The Leaky IF model incorporates a leak resistance term to make the behavior more realistic [9].

The Leaky IF model is the most commonly known current-based integrate-and-fire model. The basic model consists of a capacitor C in parallel with a resistor R driven by a current, I(t) [3]. The added resistance accounts for the leakage currents through the membrane and is modeled by:

$$C\frac{dv}{dt} + \frac{v}{R} = I(t) \tag{1.4.2}$$

Rewriting this equation and multiplying by R,

$$\tau\frac{dv}{dt} = -v(t) + RI(t) \tag{1.4.3}$$

where τ = RC is the time constant of the neuron. Equation (1.4.3) describes the dynamics of a neuron until the membrane potential reaches threshold indicating a spike has occurred (i.e. for $v < v_{thres}$). Once the potential is reset, it takes on these dynamics once more until the next spike occurs [3].

In the previous models, the synaptic input is viewed as a fluctuating current. However, a more correct physiological way to model the synaptic input is as a conductance. Increases in conductance are the physiological mechanisms that affect the integrating properties of the neuron. This is something current-based models overlook. Therefore we choose our model for membrane potential to be a conductance-based leaky IF model. In Chapter 2.1, we will discuss a conductance-based integrate-and-fire model.

CHAPTER 2

THE MODEL

To model a neuron or a population of neurons we need to keep track of the membrane potential, and changes in conductance. When modeling a population of neurons, the dynamics of the changes in conductance depend on the strength of connections to other neurons within the population as well as any external input. Being able to estimate accurate spike times is also important in correctly representing these conductance changes. Not only do we want to model membrane potential and conductance, but since the goal is to be able to match experimental brain imaging data, we also need to decide how to model calcium as well as keep track of the calcium ratios. In the following sections, we will discuss the models of membrane potential, conductance, calcium, and YFP/CFP ratios.

2.1 MODELING MEMBRANE POTENTIAL

In the previous chapter, we discussed current-based IF models. However, since we want a more physiological model, we decide to use a conductance-based IF model. The dynamics of a single integrate-and-fire, conductance-based point neuron is given by the following:

$$C\frac{dv}{dt} = -g_{leak}(v - V_r) - g_e(t)(v - V_E) - g_i(t)(v - V_I) \qquad (2.1.1)$$

where v represents the membrane potential or voltage of the neuron, g_{leak} is the leakage conductance, V_r is the rest potential of the neuron, V_E and V_I are the excitatory and inhibitory reversal potentials, and $g_e(t)$ and $g_i(t)$ are the time-dependent conductances whose dynamics will be described in section 2.2 [10]. The conductances (g_{leak}, g_e, g_i) are normalized by the membrane capacitance, C, giving them dimensions of inverse time (i.e. g_{leak}= 50/second) [20]. Once the membrane potential, v, reaches threshold, $\bar{v}$, the neuron generates a spike (fires). Immediately after the neuron fires, its membrane potential is reset to its reset potential, $\hat{v}$, which we take to be the same as the rest potential, V_r [10]. Then there is a three-millisecond refractory period until the voltage begins to integrate again. This is done to make the model more realistic since neurons cannot fire faster than once every 2 to 3 milliseconds. Once the refractory period is over, the membrane potential of the neuron once again takes on the dynamics given in equation 2.1.1.

We let the rest or reset potential, $V_r = \hat{v} = 0$, and the threshold, $\bar{v} = 1$. Since we are taking the rest and reset potential to be the same, we can consider the difference between $\bar{v}$ and $\hat{v}$ to nondimensionalize the membrane potential [10]. We then use commonly accepted values for various biophysical parameters to obtain the values for the reversal potentials: $V_E = \frac{14}{3}$ and $V_I = -\frac{2}{3}$ [9,10,20]. Now the only variable left with dimension is time, t. We can therefore rewrite equation 2.1 in the form

$$\frac{dv}{dt} = -\alpha(t)v + \beta(t) \tag{2.1.2}$$

where

$$\alpha(t) = 50 + g_e(t) + g_i(t) \tag{2.1.3}$$

$$\beta(t) = \frac{14}{3} g_e(t) - \frac{2}{3} g_i(t) \tag{2.1.4}$$

Equation 2.1.3 represents the nondimensionalized conductance of the neuron and is the inverse of an effective integration time scale [10]. Equation 2.1.4 can be seen as a difference current since it involves the difference between the excitatory and inhibitory currents [10]. We can easily extend this model for one neuron into a population of neurons by noting that each neuron within the population will follow the dynamics of equation 2.1.2. The interaction between neurons in the population comes from changes in conductance due to spikes propagating via synaptic connections with other neurons.

We now need to decide how to model the conductance. The changes in conductance of a given neuron arise from pre-synaptic, spike-induced neurotransmitters released at the neuron's synapses. In the following section, we discuss how we model changes in conductance.

2.2 Modeling Conductance

In a population of neurons, changes in conductance can occur from any external input into the neuronal population as well as from the network activity within it. Therefore, our model for time-dependent excitatory and inhibitory conductance for the j^{th} neuron in the population is given by

$$g_e^j(t) = g_{e0}^j(t) + \sum_k A_{j,k} \sum_l G_e(t - t_l^k) \tag{2.2.1}$$

$$g_i^j(t) = g_{i0}^j(t) + \sum_k B_{j,k} \sum_l G_i(t - T_l^k) \tag{2.2.2}$$

where $g_{e0}^j(t)$ and $g_{i0}^j(t)$ represent external input conductance for excitation and inhibition respectively. To calculate the changes in conductance due to network activity we have excitatory and inhibitory postsynaptic conductance functions, G_e and G_i where t_l^k and T_l^k represent the time of the l^{th} spike of the k^{th} excitatory or inhibitory neuron. The synaptic-induced conductance function is given by

$$G(t - t_{spike}) = \frac{(t - t_{spike})}{\tau^2} e^{-(t - t_{spike})/\tau} \Theta(t - t_{spike}) \tag{2.2.3}$$

G_e and G_i are given by equation 2.2.3 with corresponding time constants τ = .001 and τ = .002 [10]. The function, $\Theta(t)$, denotes the Heaviside Function which is zero when t - t_{spike} is negative and one when t - t_{spike} is positive.

We derive the equation above from the following system of differential equations:

$$\begin{aligned} \frac{dg}{dt'} &= -\frac{g}{\tau} + \frac{w}{\tau} \\ \frac{dw}{dt'} &= -\frac{w}{\tau} + \frac{\delta(t')}{\tau} \end{aligned} \tag{2.2.4}$$

where $\delta(t')$ is the Dirac delta function[1]. Solving this system of equations and taking $t' = t - t_{spike}$, we get the following solutions:

[1] The Dirac Delta Function is a distribution that is zero everywhere except for one point, where it is infinitely big. The integral of a delta function from $-\infty$ to ∞ is 1.

$$g(t-t_{spike}) = C_1 e^{-(t-t_{spike})/\tau} + C_2 \frac{(t-t_{spike})}{\tau} e^{-(t-t_{spike})/\tau} + \frac{(t-t_{spike})}{\tau^2} e^{-(t-t_{spike})/\tau} \Theta(t-t_{spike})$$
$$w(t-t_{spike}) = C_2 e^{-(t-t_{spike})/\tau} + \frac{e^{-(t-t_{spike})/\tau}}{\tau} \Theta(t-t_{spike}) \quad (2.2.5)$$

Starting at time zero ($t_0 = 0$) such that $C_1 = g(t_0) = 0$ and $C_2 = w(t_0) = 0$, we see the initial change in conductance follows the dynamics of equation (2.2.3). During the simulation of the model, we keep track of both equations in 2.2.5 as we integrate through time to accurately measure changes in conductance.

As seen in the models for membrane potential and conductance it is very important to precisely estimate the time of a spike. The dynamics of both variables highly depend on the spike time. In the following section, we will discuss a fourth order and second order Runge-Kutta method for numerically simulating this system of equations.

2.3 Runge-Kutta Method and Estimating Spike Times

Runge-Kutta (RK) is an algorithm designed by Carl Runge and Wilhelm Kutta. RK methods are used to numerically integrate ordinary differential equations over time. It was built to resemble the Taylor Series Method without having to determine any derivatives. The second order Runge-Kutta (RK2) and fourth order Runge-Kutta (RK4) methods are the most common forms used in simulations [18].

The RK2 method requires two function evaluations at each time step and is accurate up to two terms in the Taylor series expansion. Therefore, it has an error of $\vartheta(h^3)$. Given an initial value problem,

$$x'(t) = f(t,x)$$
$$x(t_0) = x_0 \tag{2.3.1}$$

a RK method is used to numerically approximate its solution. The formula for RK2 is as follows:

$$x(t+h) = x(t) + \frac{h}{2}(k_1 + k_2) \tag{2.3.2}$$

where

$$k_1 = f(t,x)$$
$$k_2 = f(t+h, x+k_1)$$

For each fixed time step h, the second order method uses a trial step at the midpoint, and then it uses the values of t and x at the midpoint to make the real step over the interval to estimate $x(t+h)$. The trial step cancels out the lower order terms, resulting in an error of $\vartheta(h^3)$ [18]. The second order method is fast in computation but may lack in precision for many simulated models. Therefore, higher order Runge-Kutta methods have been developed.

The most commonly used RK method is the fourth order method. The fourth order method is obtained at the cost of four function evaluations, but has error in the order of $\vartheta(h^5)$. This means that the improved formula agrees up through four terms of the Taylor series expansion. The balance between costs of computation versus precision per time step is the reason that RK4 is most often implemented.

At each time step, there are three trial steps. The two extra trial steps (versus RK2) provide a better approximation to the behavior surrounding the midpoint of the interval. We have the following formula for the RK4 method:

$$x(t+h) = x(t) + \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{2.3.3}$$

where

$$\begin{aligned} k_1 &= f(t,x) \\ k_2 &= f(t + \tfrac{1}{2}h, x + \tfrac{1}{2}k_1) \\ k_3 &= f(t + \tfrac{1}{2}h, x + \tfrac{1}{2}k_2) \\ k_4 &= f(t + h, x + k_3) \end{aligned}$$

The method is relatively simple and has a higher precision than the RK2 method [18]. Now that we have the formulas, we can use RK methods to integrate the voltage of a neuron.

Standard RK methods will calculate the membrane potential at the beginning of each time step. However, this can cause numerical errors to the membrane potential after a spike since the neuron may have fired at any point within the time step. To avoid such errors, we follow from Shelley and Tao to improve the typical RK2 and RK4 methods by using linear (RK2) and cubic (RK4) interpolants to better approximate firing times between the time steps [10].

Using equation (2.1.2) for membrane potential, a single step of RK2 would look like:

$$\begin{aligned} v_{n+1} &= v_n + \frac{h}{2}(k_1 + k_2), \\ k_1 &= f(t, v_n) = -\alpha_0 v_n + \beta_0, \\ k_2 &= f(t + h, v_n + k_1 h) \\ &= -\alpha_1[v_n + h(-\alpha_0 v_n + \beta_0)] + \beta_1 \end{aligned} \tag{2.3.4}$$

where $\alpha_0 = \alpha(t),\ \alpha_1 = \alpha(t+h)$ and $\beta_0 = \beta(t),\ \beta_1 = \beta(t+h)$ [10]. If the voltage at v_{n+1} is less than the threshold, we use the already calculated v_{n+1} in the next time step. If the voltage at v_{n+1} is greater than the threshold value, then we know the neuron has fired at some point over the time step. This means at the next time step, we will start from rest. However, we want to be more accurate when estimating the actual spike time. We accomplish this during a time step by using a linear interpolant since RK2 is only of second order:

$$v(t) = v_n + \frac{(v_{n+1} - v_n)}{h} t \tag{2.3.5}$$

assuming, without loss of generality, t_n=0 [10]. We know a spike occurs when $v(t)=v_{thres}$. So we estimate the spike time, t_{spike}, by solving equation (2.3.5) for

$$v(t_{spike}) = v_{thres} = v_n + \frac{v_{n+1} - v_n}{h} t_{spike} \tag{2.3.6}$$

whenever v_{n+1} is above threshold. Having found t_{spike}, we can use it to find a new v_n estimate and new v_{n+1} estimate to more accurately estimate the post-synaptic membrane potential. With one RK2 step we can recalculate the membrane potential after a spike. This continues for each time step throughout the simulation [10].

Given the formula for the RK4 method (2.3.3) we get the following

$$
\begin{aligned}
v_{n+1} &= v_n + \frac{h}{6}(k_1 + k_2 + k_3 + k_4) \\
k_1 &= f(t_n, v_n) = -\alpha_0 v_n + \beta_0 \\
k_2 &= f(t_n + \tfrac{1}{2}h, v_n + \tfrac{1}{2}k_1 h) \\
&= -\alpha_{1/2}(v_n + \tfrac{1}{2}k_1 h) + \beta_{1/2} \\
k_3 &= f(t_n + \tfrac{1}{2}h, v_n + \tfrac{1}{2}k_2 h) \\
&= -\alpha_{1/2}(v_n + \tfrac{1}{2}k_2 h) + \beta_{1/2} \\
k_4 &= f(t_n + h, v_n + k_3 h) \\
&= -\alpha_1(v_n + k_3 h) + \beta_1
\end{aligned}
\tag{2.3.7}
$$

where h is the time step, $\alpha_0 = \alpha(t_n)$, $\alpha_{1/2} = \alpha(t_n + \frac{1}{2}h)$, and $\alpha_1 = \alpha(t_n + h)$. The β's follow similarly. For a fourth order method, we need a cubic interpolant instead of a linear interpolant. Therefore, we use a cubic Hermite polynomial as the interpolant [10]. A Hermite polynomial is used instead of a Newton polynomial since we already know the approximations of v_n and v_{n+1}, and have their derivatives using the α's and β's from the RK4 method

$$
v_n' = f(t_n, v) = -\alpha_0 v + \beta_0 \tag{2.3.8}
$$

$$
v_{n+1}' = f(t + h, v_{n+1}) = -\alpha_1 v_{n+1} + \beta_1 \tag{2.3.9}
$$

Hermite polynomial interpolation with two data points and two derivatives may have at most degree 3, which will give us the cubic interpolant we want for our simulation,

$$
v(t) = v_n + v_n' t + \left[\frac{3(v_{n+1} - v_n) - h(2v_n' - v_{n+1}')}{h^2}\right] t^2 + \left[\frac{-2(v_{n+1} - v_n) + h(v_n' + v_{n+1}')}{h^3}\right] t^3 \tag{2.3.10}
$$

We want to solve (2.3.10) for $v(t_{spike})=v_{thres}$ so we numerically determine the zero of the function corresponding to t_{spike} (using the fzero command in MATLAB)[1]. From here, we once again get a new estimate for v_n and then use our RK4 method to find the new estimate for v_{n+1}. This method is used for each time step where a spike has occurred [10].

It is possible to use the second or fourth order RK method in the simulation of a population of neurons and have a reasonable estimate for the actual spike time. In our simulations we choose to use the fourth order RK method.

The Runge-Kutta algorithm is also used to integrate the model for calcium. In the following section, we will discuss why and how we model calcium and its ratios within the cell.

2.4 Calcium and Ratios

The human brain is considered to be one of the most complex objects to study [13]. It is made up of approximately 100 billion neurons and 100 trillion synapses with tens of thousands of connections to surrounding neurons [12]. Luckily for researchers, the neural activity in the human brain is similar to the neural activity in the brains of other animals with less complex brain structures. One of these animals is the Zebrafish. The Zebrafish brain has around 10,000 neurons, which is significantly smaller than the human brain, and develops quickly (growing from a single cell to larva in less than 3

[1] The fzero command uses a combination of bisection, secant, and inverse quadratic interpolation methods to find the zero associated with the value of the spike time.

days) making it useful for studying brain activity. Zebrafish are also translucent for certain wavelengths of light allowing researchers to view what is going on in the brain [13].

To induce seizures, researchers bathe the Zebrafish in a solution containing a drug called Pentylenetetrazole (PTZ), which is a $GABA_{A,C}$ antagonist. GABA is the main inhibitory neurotransmitter and $GABA_{A,C}$ is the receptor site for these neurotransmitters. By blocking these receptor sites, PTZ reduces the amount of inhibition in the brain causing runaway bursting activity.

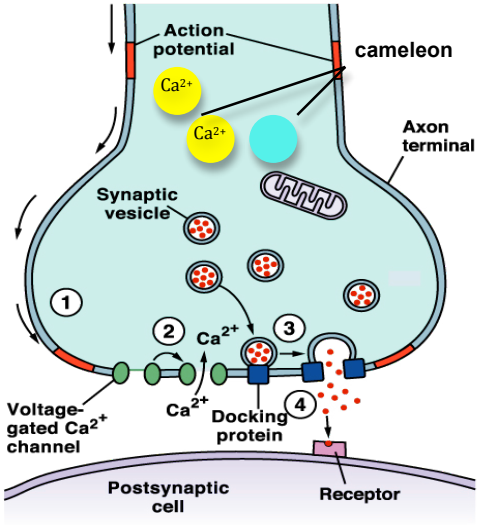


FIGURE 2.1 SYNAPTIC TRANSMISSION. Shows release of calcium through the voltage-gated calcium channels during an action potential. It also shows the transmission of neurotransmitters over the synaptic cleft [11].

So why do we image calcium? As an action potential or spike moves across the axon terminal, it opens voltage-gated calcium channels. Calcium is then released into the terminal allowing synaptic vesicles containing neurotransmitters to attach themselves to the terminal membrane. This allows the neurotransmitters to travel

across the synaptic cleft to activate receptors on the postsynaptic membrane [13]. Hence, calcium is a by-product of neuronal firing. An influx of calcium is therefore a good indicator of neuronal firing.

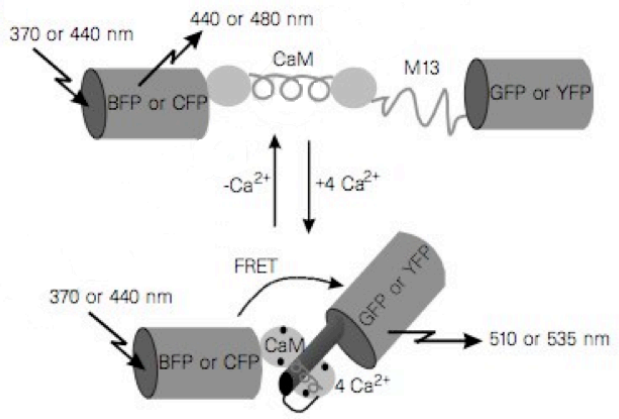


FIGURE 2.2 USING FRET TO MEASURE CALCIUM. Diagram shows how FRET between CFP and YFP can measure calcium concentration [14].

To visualize neural activity during a seizure, researchers use a genetically engineered protein called Cameleon. Cameleon serves as a fluorescent indicator for calcium. It is based on the green fluorescent protein (GFP), which emits green fluorescence, and calmodulin (CaM), a calcium binding protein [14]. More specifically, we use Cameleon YC2.1, which is a molecular complex whose structure is based on two GFP mutants, cyan fluorescent protein (CFP) and a yellow fluorescent protein (YFP) separated by CaM and a calmodulin-binding peptide, M13 [16]. Cameleon YC2.1 is created to detect increases in calcium by Förster resonant energy transfer (FRET). If there is no bound calcium in the cell, cyan fluorescence will be emitted under blue-violet light (440 nm). However when calcium ions are bound (i.e. calcium is in the cell), CaM wraps around the binding peptide bringing the CFP and YFP closer together (see Fig 2.2). This increases the FRET between the two fluorescent proteins. Under blue-violet

light, the cell will emit a yellow fluorescence in the presence of calcium (see Fig 2.2). Therefore, the degree of FRET in the cameleon can be used as a measurement of calcium in the cell [14,15,16]. When there is an increase of calcium in the cell, as seen in Figure 2.3, there is an increase in YFP and a decrease in CFP. And the maximum ratio occurs during an increase of calcium.

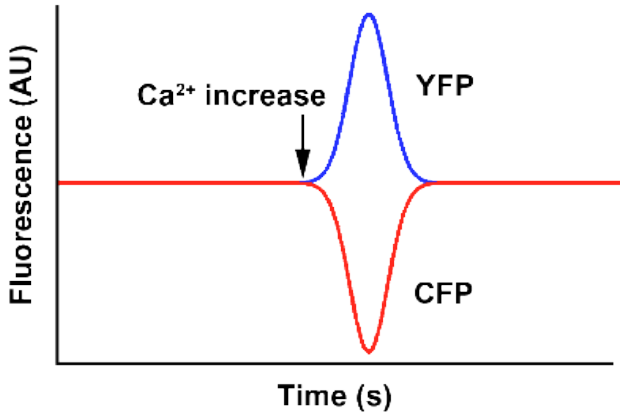


FIGURE 2.3 Graph of Calcium ratios during an increase of calcium in the cell. (Graph by Andrew Sornboger)

During a seizure there is an excess of calcium due to the rapid neuronal firing. Therefore it is possible to use these genetically engineered fluorescent proteins to observe seizure-related calcium activity in the brain. It also provides a ratiometric measurement of the calcium. Since experimenters use these methods to understand seizures from the brain imaging point of view, it is also important to predict calcium concentrations and YFP/CFP ratios.

We model calcium concentration within a cell, c, with a simple linear differential equation,

$$\frac{dc}{dt} = -\tau_c^{-1} c + \phi_c I_c \tag{2.4.1}$$

The constant τ_c represents the time constant that regulates the decay of calcium transients. I_c is the calcium current across the membrane and the constant ϕ_c scales the amplitude of the calcium transients during a spike [3]. Therefore, fitting this equation to experimental results [17], the resulting equation is given by

$$\frac{dc}{dt} = -\frac{1}{2} c + 10^{-5} \delta(t) \tag{2.4.2}$$

The emission ratio, R, of cameleon is modeled by the equation

$$R = \frac{c r_{\max} + K_d' r_{\min}}{K_d' + c} \tag{2.4.3}$$

where c is calcium, $K_d' = 10^{-6.5}$ represents the apparent dissociation constant, and $r_{\max}$ = 1 and $r_{\min}$ = 0 are the maximum and minimum values of the emission ratio. Equation 2.4.3 is based on experimental results from Miyawaki, [15].

Now that we have our models for the simulation, we need to decide how to connect the neurons in the population. There are several ways to view these connections. In the next chapter we will discuss two ways we constructed the connection matrices for a population of neurons.

CHAPTER 3

THE CONNECTION MATRIX

To model a population of neurons, it is important to consider how the neurons within the population are connected to each other. The main way neurons communicate is via synapses. So, with a conductance-based integrate-and-fire model these synaptic connections within the network and their strengths drive the membrane potential for each neuron to spike. We describe these connections in the network through a connection matrix. In the following sections we will describe two different ways we modeled the connection matrix.

3.1 ORIGINAL CONNECTION MATRIX: RANDOM WEIGHT CONSTRUCTION

The results from experimental imaging data suggest the movement of a seizure in the Zebrafish brain as wavelike, moving from one part of the brain to the next. Hence, the original connection matrix for the model is based on two populations of neurons interacting with each other. In the connection matrix, the rows represent neurons and the columns represent the connections that a neuron has with other neurons in both populations (see Figure 3.1). A main goal of the simulation is to produce a wave of spikes from the first population to the second population.

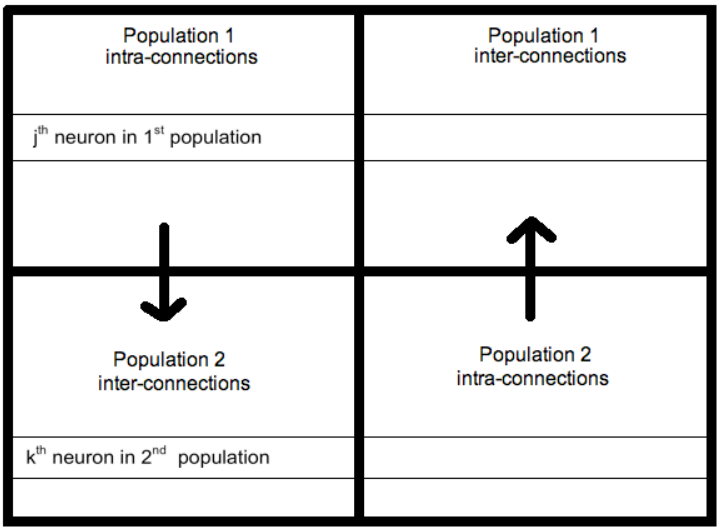


FIGURE 3.1 CONNECTION MATRIX WITH TWO POPULATIONS. Visualization of connection matrix. Each row represents a neuron. The columns represent the connection to the other neurons within the two populations.

Using this connection matrix, we simulated two excitatory neurons interacting with each other as well as two populations of fifty excitatory[1] neurons interacting with each other. In the simulation of the two populations, we assume we have more intra-connections than inter-connections in our model. This is represented by having each neuron randomly connected to 10 neurons within their population and to three random neurons in the other population (see Figure 3.2). Also, no neuron is connected to itself. Finally, five neurons in the middle of the first population receive external input allowing them to spike on their own. Hence, a neuron in the second population can only spike in response to the excitation in the first population.

[1] We assume there is no inhibition in the system since most of the inhibition has been wiped away in the Zebrafish by PTZ.

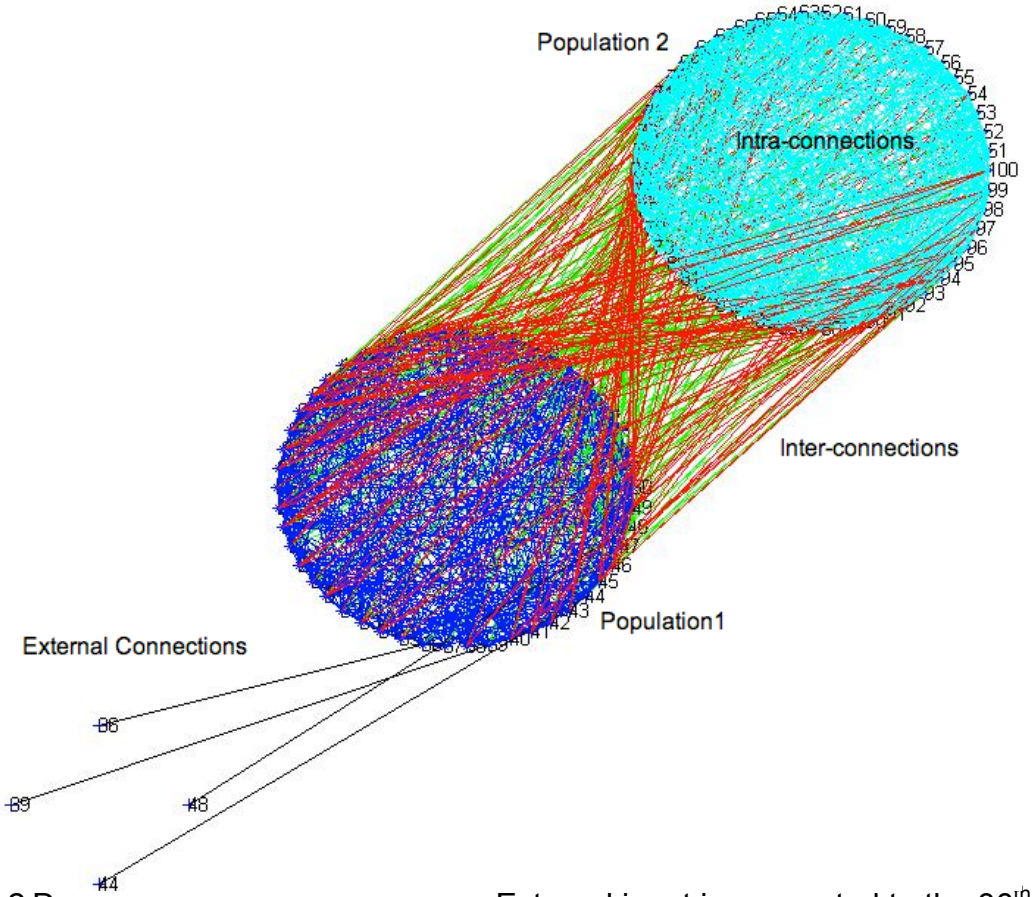


FIGURE 3.2 DIAGRAM OF CONNECTION MATRIX. External input is connected to the 36th, 39th, 44th, and 48th neuron in the first population. Connections within the first population are blue and the connections from the first population to the second are red. Connections within the second population are cyan and the connections from the second population to the first are green.

The strengths of synaptic connections are also based on whether the connection is within one population or between two populations. In our model, we assume the intra-connections are stronger than the inter-connections. Also, due to the intrinsic variability of synapses, we assume these synaptic strengths follow a random uniform distribution.

After reviewing the results using this connection matrix, we were having trouble finding a combination of synaptic strengths to produce the results similar to those seen in experimental data (see Chapter 4.2). Therefore, we began to look at a more realistic

spatial arrangement for the connection matrix. This connection matrix and the spatial grid are described in the next section.

3.2 Final Connection Matrix: Spatially Organized

The previous connection matrix allowed for too much excitation within the system to get the wave-like pattern we were looking for in the simulation. To improve our model, we decided to add some inhibition into the system and change the structure of our connection matrix. Following Ursino and La Cara ([19]), we considered the arrangement of our population of neurons in terms of a spatial grid or lattice where each element represents a neuron. Once we determined the connections and their strengths, we constructed a new connection matrix based on the spatial grid.

For the synaptic connections, we assume a classical Mexican-hat disposition where the spatial extension of the inhibitory synapses has been taken greater than that of the excitatory synapses. In terms of strength of connections, we assume it decreases with distance. And as in the previous model, to account for the natural variability of the synapses, we assume the synaptic strengths follow a random uniform distribution between a maximum and minimum value [19]. Therefore our equations to signify the synaptic connections to the ij^{th} neuron by excitatory (Ex) and inhibitory (In) synaptic inputs are given by

$$Ex_{ij} = \sum_{l,m=1}^{N} W_{ex0}\rho_{ij}e^{-d_{ij,lm}^2/\sigma_{ex}^2} \tag{3.2.1}$$

$$In_{ij} = \sum_{l,m=1}^{N} W_{in0}\rho_{ij}e^{-d_{ij,lm}^2/\sigma_{in}^2} \tag{3.2.2}$$

$$d = \sqrt{(i-l)^2 + (j-m)^2} \tag{3.2.3}$$

where W_{ex0} and W_{in0} are the strength of the excitatory and inhibitory synapses; σ_{ex} and σ_{in} are the standard deviations, which imitate the decrease of strength due to increases in distance; ρ_{ij} is a random variable with uniform distribution between 0.5 and 1.5; and d represents distance between neurons [19]. We also assume the spatial grid has periodic boundary conditions (see figure 3.3).

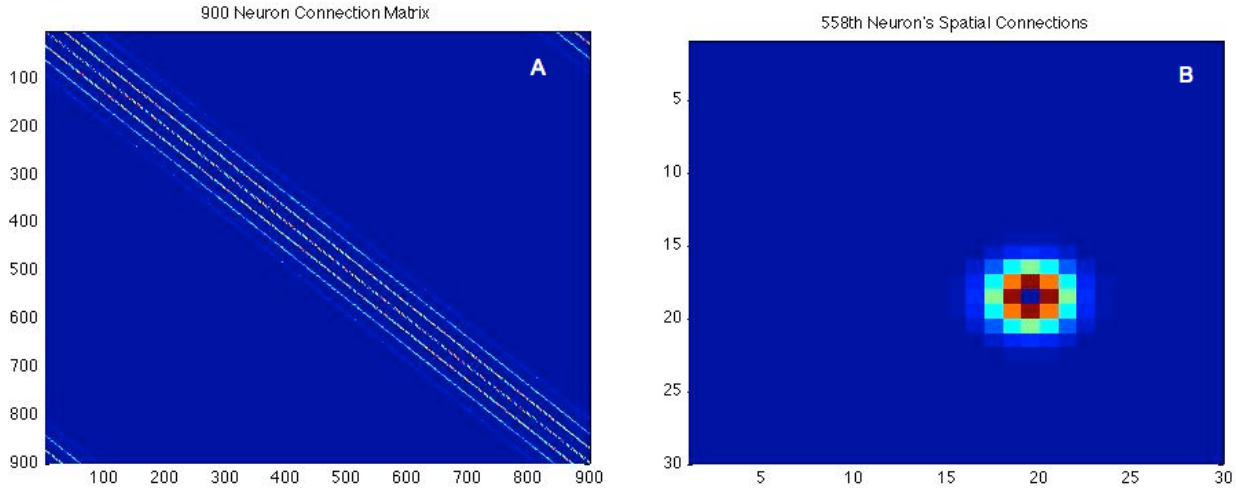


FIGURE 3.3 VIEWS OF CONNECTION MATRIX AND SPATIAL GRID. A) 900-neuron connection matrix; B) 558th neuron's connections on spatial grid

With this outline for our spatial grid, we simulate a population of 900 neurons using a 30 x 30 interconnected network of neurons (see Fig 3.3). Throughout part of the simulation, a 3 x 3 cluster of neurons in the center of the grid receive external input allowing these neurons to spike. Noise can also be added into the system [19].

The construction of the spatial grid and the strengths of the synaptic connections determine the impact synaptic-induced conductance changes have on other neurons within the connection matrix. In the next section we will see just how these connections affect the membrane potential of each neuron.

CHAPTER 4

THE RESULTS

In the following sections we will discuss results from various simulated models. First we will look at two neurons. We will see how one neuron affects the voltage and conductance levels of another neuron. Then we will consider two populations of neurons with the original connection matrix presented in section 3.1. Finally, we will discuss results from our larger population of neurons with a connection matrix based on the spatial grid presented in section 3.2.

4.1 TWO NEURONS

We wanted to understand how the action potentials of one neuron affected the voltage and conductance of another neuron through its synaptic connections. In our simulation, we assumed the first neuron had only a constant external input of 14 hertz. The strength of this input allowed Neuron One to spike on its own. Neuron Two had no external input. It had a one-way connection to the Neuron One. Neuron Two's voltage and conductance changed based on the number of spikes from Neuron One, but it did not contribute to the changes in membrane potential of Neuron One.

The synaptic connections between neuron 1 and neuron 2 were both excitatory and inhibitory and were based on the spatial grid (see Chapter 3.2) with a distance of

one. The excitatory synaptic strength was 1.5 and the inhibitory synaptic strength was .6. After 200 milliseconds, Figure 4.1 shows a set of these results. We saw Neuron One spike consistently given the constant conductance. Once Neuron One spiked, it then caused Neuron Two to spike. We saw the dynamics of the voltage of the postsynaptic neuron vary from that of Neuron One. This simulation gave us a view of the impact synaptic-induced conductance changes have on connected neurons.

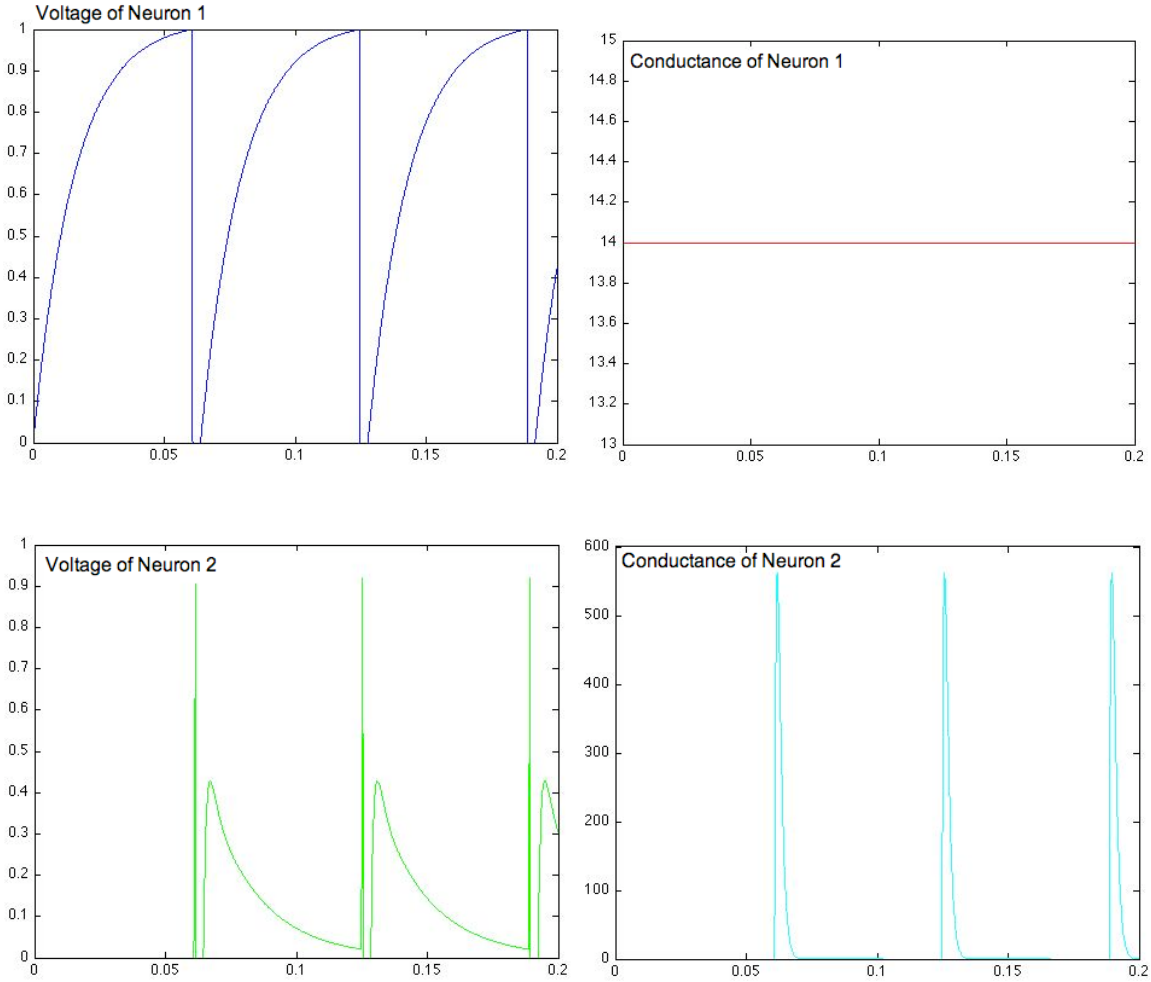


FIGURE 4.1 TWO-NEURON SIMULATION. Voltage and Conductance of neuron 1 and neuron 2 with W_{ex0}=1.5, W_{in0}=.6. Neuron 1 has g_{ext}=14 hertz. Neuron 2 is distance 1 away from neuron 1.

It is evident that the strength of the synaptic connection between two neurons influenced the impact of the conductance changes. Since the strength of connection was based on distance, in our next simulation we considered two neurons that were

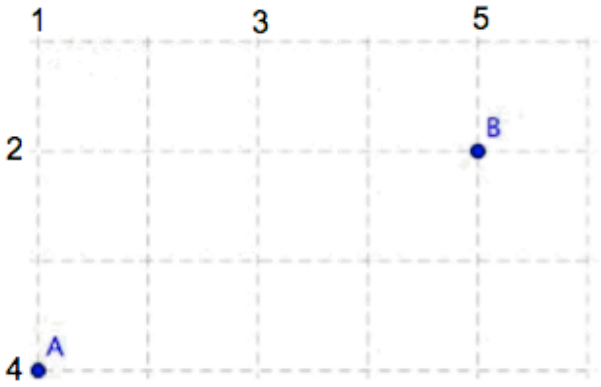


FIGURE 4.2 SPATIAL GRID FOR TWO NEURONS: Neuron A and B.

farther away from each other. Neuron A and Neuron B are seen on the grid in Figure 4.2. We asked the question: How will this affect the conductance and voltage of neuron B? We discovered that an increase in distance (i.e. decrease in connection strength) reduced the impact of a spike in neuron A on the membrane potential of neuron B. Looking at figure 4.3, we see that neuron B does not even fire. The conductance changes due to the synaptic connection were not strong enough to cause the membrane potential to reach threshold.

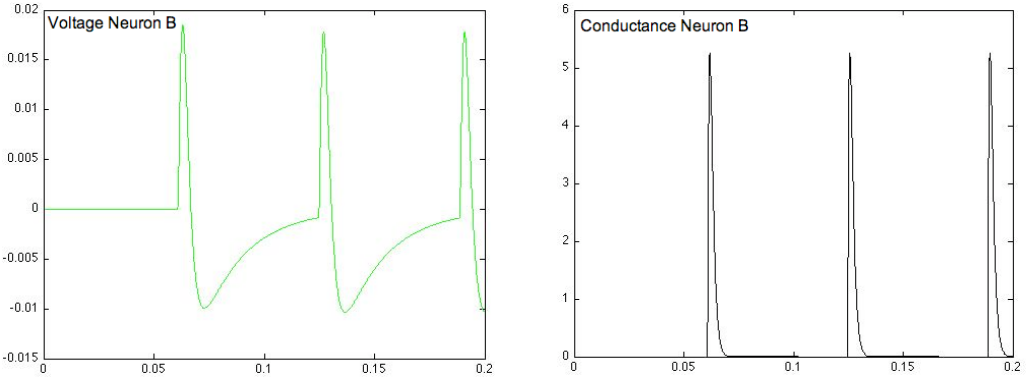


FIGURE 4.3 VOLTAGE AND CONDUCTANCE OF TWO NEURONS: Neuron B with distance 4.42 from Neuron A.

Now that we have seen how the spiking of one neuron influences the membrane potential of another, we wanted to simulate how a population of neurons influenced another population of neurons. In the next section we will discuss the results from the two-population model.

4.2 Two Populations of Neurons

Since we've seen how two single neurons communicate, the next step is to see how two populations communicate. The imaging of calcium suggests that a seizure moves through parts of the brain in a wave-like pattern. And from this data firing rate frequencies are also predicted. In this simulation, we decided to think of the different parts of the brain as different populations of neurons. With our model, we attempted to simulate a similar wave pattern and produce similar firing rate frequencies and calcium levels as seen in the experimental results.

In the experiments with Zebrafish, PTZ wipes out the inhibition in the brain so we only considered populations of excitatory neurons. We simulated 400 neurons with 200 neurons in each population. We ran the simulation for one second with a time step of .1 milliseconds. The first population had five neurons that received a constant external input of 20 hertz. The remaining neurons would produce spikes due to their synaptic-connections to these neurons. The connection matrix followed the model in Chapter 3.1 and no neurons in the second population had a direct connection to the five neurons in population 1 with the external input (see Appendix for MATLAB code).

In our first example we examined the effect of varying the intra-connection strengths for the first and second population. The inter-connection strength for population 1 and 2 was equal to .3. The external input was turned on for one thousandth of a second and then turned off for the remaining time period. First, we were looking for a low threshold in which the spiking activity of the stimulated neurons in population 1 did not propagate toward surrounding neurons. We found this threshold with intra-connection strength of .1 as seen in figure 4.4. The simulated neurons fired but did not cause other neurons to spike. Since there was some randomness in the connections of the network, extreme differences in behavior can sometimes be observed. However, these parameters on average maintained the observed behavior as a threshold.

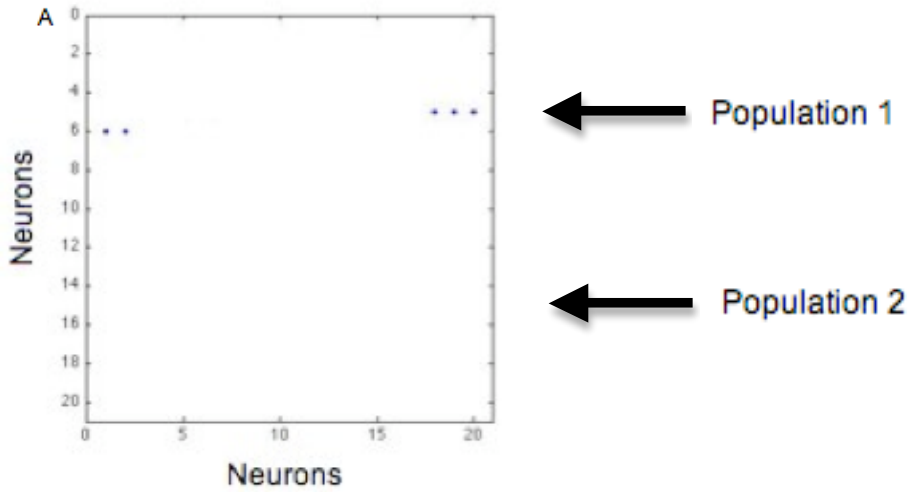


FIGURE 4.4 TWO POPULATION MODEL: FIRING AND SVD ANALYSIS. A) Population firing at .609 sec. The five neurons with external input fire and then dissipate. They do not affect the surrounding neurons.

We also wanted to see how an increase in intra-connection strengths changed the movement of the propagating waves. With an increase of intra-connection strengths for both populations to .2, we saw that the stimulated neurons do affect the firing rate of

surrounding neurons in population 1. Then, once population 1 begins firing, it caused a few of the neurons in population 2 to fire. Once population 2 had intra-firing, it influenced more of its neurons to fire until it died out. This type of behavior continued throughout the simulation.

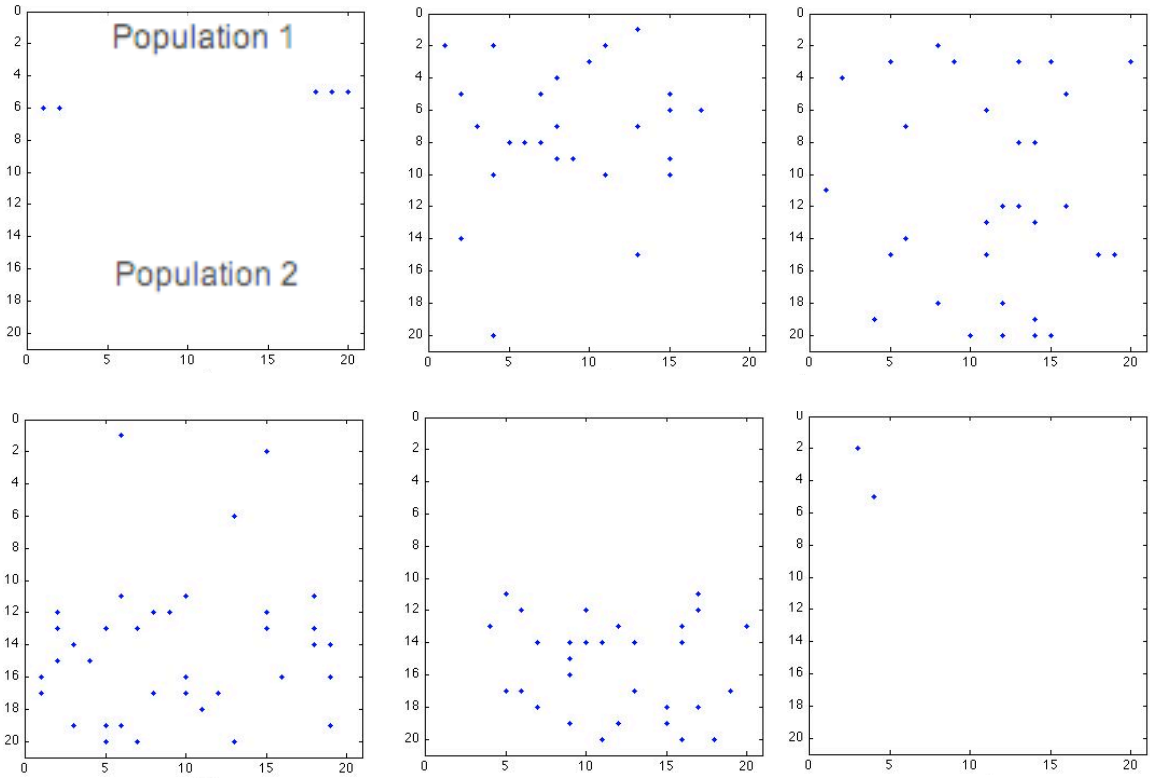


FIGURE 4.5 TWO POPULATION MODEL: FIRING PROPAGATION THROUGH POPULATIONS. Glimpses of the propagation of neuronal firing over time from population 1 (first 10 by 20) to population 2 (second 10 by 20) with intra-connection strengths of .2.

To understand the data more clearly, we calculated the singular value decomposition (SVD). Finding the SVD provides us with the eigenvectors that contribute the highest variance in the network. From this, we can eliminate a large portion of the data and focused on the most influential eigenvectors. An SVD lists the eigenvalues and corresponding eigenvectors in order from producing the most variance to the least variance. In figure 4.6A, we plotted the corresponding eigenvalues and determined where to truncate the data. Taking a look at just the first eigenvector, we graphed the

firing rate frequency over time (4.6B). The graph showed strong low frequency firing rates. The base frequency was around 75 hertz. In experimental data, researchers predict firing rate frequencies around 50 – 60 hertz. Therefore, in this example our results were similar to those from the brain imaging data.

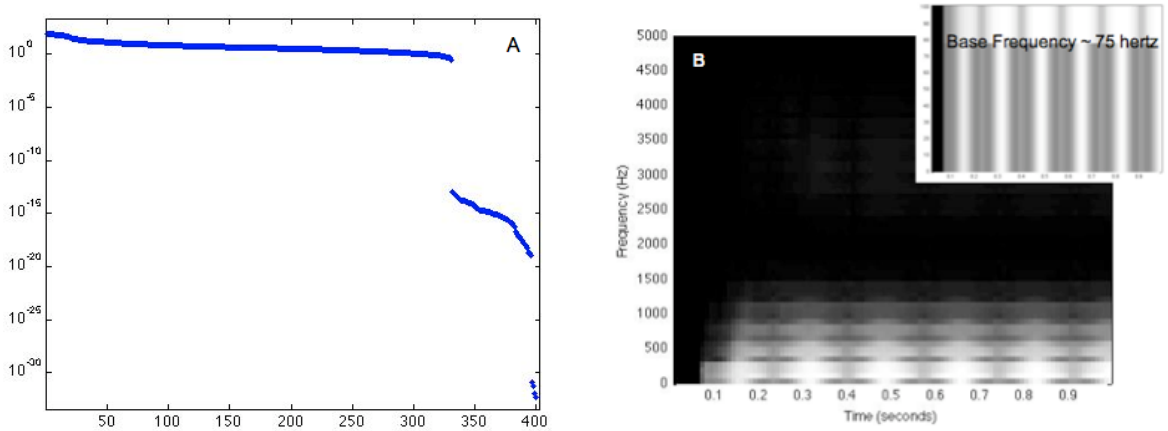


FIGURE 4.6 TWO POPULATION MODEL: EIGENVALUES AND FIRING RATE FREQUENCY. A) Log scale for y-axis vs. eigenvalues plot of singular valued decomposition (SVD). Allows researcher to identify the eigenvectors involved in the highest variation in the system. B) Grayscale plot of firing rate frequency of the first eigenvector over time. Shows low frequency firing rates.

Since we had similar firing rate frequencies, the next data we reviewed was the calcium and ratios data to see how it compared to what was seen in the experimental data. We expected to see levels of calcium to be around 10^{-7} to 10^{-5} and the maximum emission ratio of cameleon to be between 40 and 60 percent. Figure 4.7 shows the results from the simulation. In 4.7A we see that the recorded maximum ratios for both populations were up to around 45 percent by the end of the simulation. We took the SVD of the calcium data and analysis showed that roughly the first seven eigenvalues caused the most variation in the system. Due to space, we present the spatial and time series data for the first four eigenvectors in 4.7B-E.

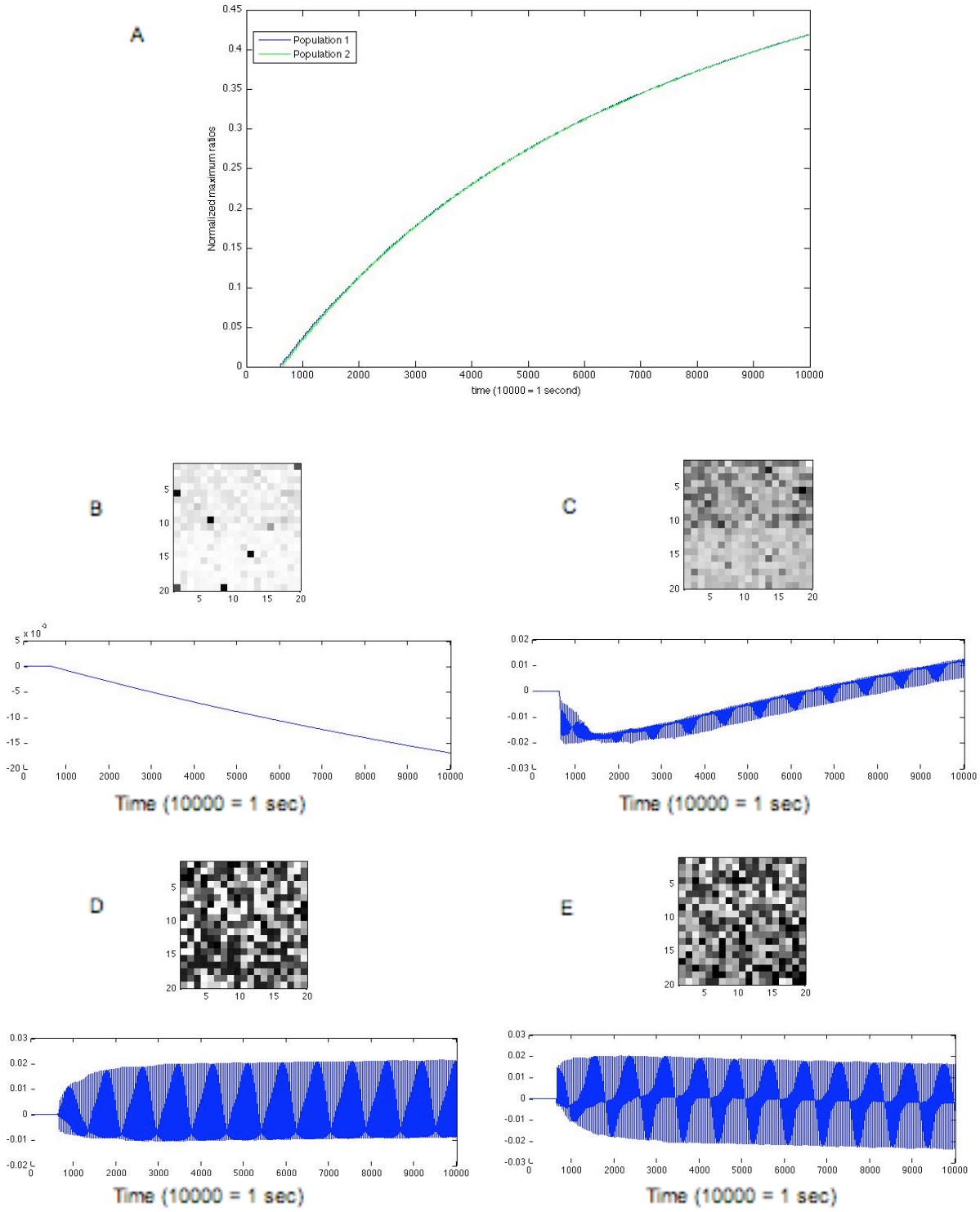


FIGURE 4.7 TWO POPULATION MODEL: CALCIUM AND YFP/CFP RATIO LEVELS A) Maximum Ratios of population 1 and population 2 over time. B-D) Spatial and time series data for the first four eigenvectors over time. These eigenvectors cause the most variation in the network.

When we continued to pump up the intra-strengths of the neurons we saw more structure to the propagation of spiking activity from population 1 to population 2. Along with this structure, however, we got higher frequency of firing rates (see figure 4.8 and 4.9). In these simulations, we also lengthened the time of external input pumping into the system, however, we did not see significantly different results (time period for simulation is one second). Therefore, these results are not presented in figures.

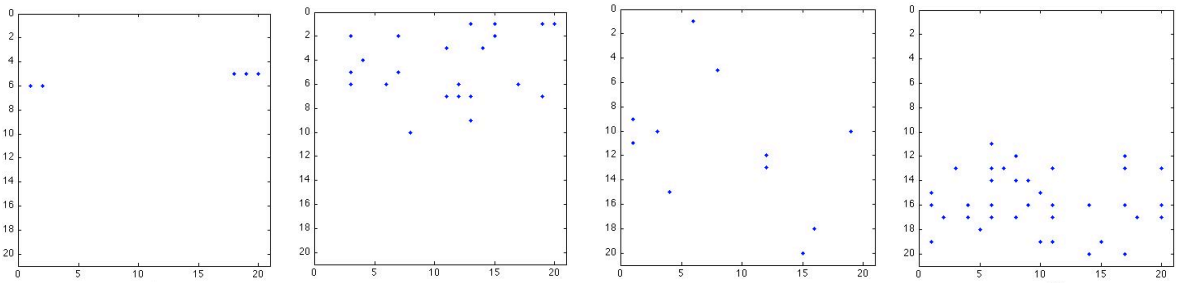


FIGURE 4.8 TWO POPULATION MODEL: FIRING PROPAGATION THROUGH POPULATIONS. Glimpse of propagation of spiking activity with intra-connection strength of .5.

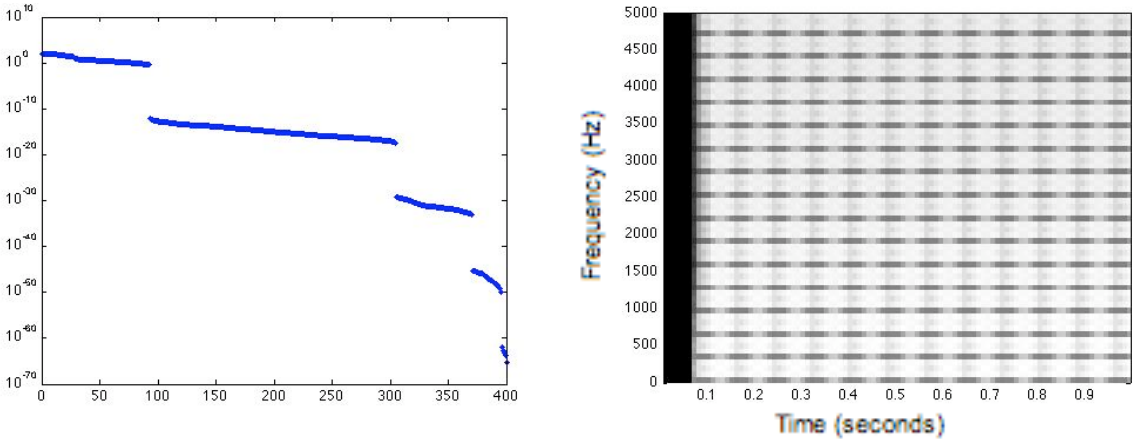


FIGURE 4.9 TWO POPULATION MODEL: EIGENVALUES AND HIGH FIRING RATE FREQUENCY. A) Log plot of eigenvalues. Shows most important eigenvalues to the system are the first 28 eigenvalues and corresponding eigenvectors. B) Frequency over time plot of first eigenvector. Shows high frequency firing rates. (Similar plot for the other 27 most influential eigenvectors.)

Given the two-population model, we did see the oscillation of spiking activity from one population to the next. However this oscillation was much faster than seen in experimental results. Also, there was too much excitation in the system. Once the first population began firing, the whole network continued to fire no matter what time the external input was turned off. Therefore, some inhibition is needed. It was also difficult to find appropriate intra- and inter- connection strengths to closely match results from experimental data and maintain low frequency firing rates.

In this model we were not considering connections with any spatial organization. We found a paper by Ursino and LaCara ([19]) where a spatial structure of connections was used that produced wave patterns with low firing rate frequencies. We therefore changed our focus to this model and its results will be discussed in the following section.

4.3 Network of Neurons Spatially Connected

In this section we present results from a set of simulations using the spatially connected network of neurons presented in Chapter 3.2. In the previous section we were not considering space when formulating the synaptic connections of the neurons. We felt this was important to consider when looking at the propagation of spiking activity through a neuronal network. We also needed to include some inhibition into the system.

In this set of simulations we looked for a low threshold in which the stimulated neurons did not cause sustaining spiking activity of surrounding neurons. Once we

located the threshold, we concentrated on the increase of excitation strength, W_{ex0}, and varying the length of time external input was on during the simulation. Both the strength of inhibition and external input were kept fixed. Our current simulation did not contain Gaussian noise, but it could be added to the system if desired.

For the following results, we used a neuronal network of 900 neurons with inhibitory strength, W_{in0}, equal to 0.2, and input conductance, g_{ext}, equal to 14 hertz. The simulation ran for two seconds with a time step of .1 milliseconds. We varied the length of time (.1 sec, .3 sec, .5 sec, .7 sec, .9 sec) the input conductance was turned on and the strength of excitatory synapses (.2, .3, .4, .5, .6). We found our lower bound when W_{ex0} = 0.2. With this parameter (depending on when the input was turned off) the stimulated neurons initiated minimal spiking activity within the network. Once the input was turned off, the spiking activity diminished. After finding this threshold, we concentrated on the behavior of the network with increased excitation. In the following discussion we will present findings with parameters W_{ex0} = 0.4 and input time = .9 seconds. The results using other parameters followed similar behavior.

First we observed the general spiking activity of the network. In figure 4.10 we highlight interesting activity that occurred throughout the two-second simulation. In 4.10A we see the progression of the initial spiking behavior for the network over .08 seconds. The nine neurons in the center of the frame first fired at .609 seconds. Once these neurons fired, the surrounding neurons were also stimulated to fire. After this firing took place, there was enough excitation in the system (and regular input from stimulated neurons) to maintain random firing within the network (see 4.10B).

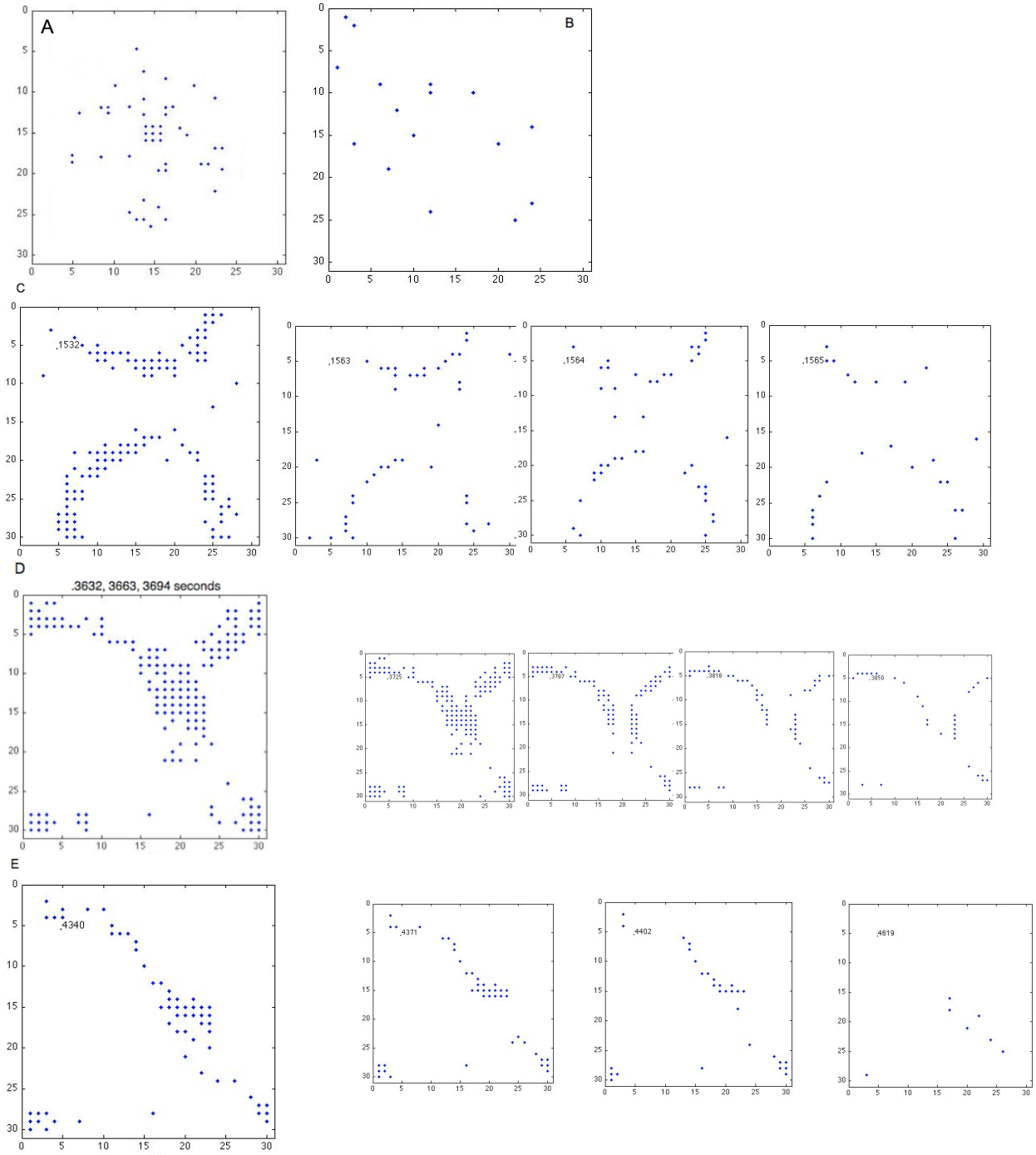


FIGURE 4.10 TWO-POPULATION MODEL: EIGENVALUES AND FIRING RATE FREQUENCY
A) Glimpses of initial spiking at .609 seconds through .683 seconds. B) Random firing throughout population. C) First circular structured pattern of firing at .1532 seconds. Similar pattern occurring at .1563, .1564, .1565, and .1600 seconds with less strength. D) Stronger circular structured firing pattern occurring at .3632, .3663, .3694 seconds. Similar pattern occurring at .3725, .3787, .3818, .3850 with less impact. E) Last structured firing pattern occurring at .4340 seconds. Similar pattern occurring at .4371, and .4402. As seen in last frame, spiking activity restricts itself to small population within the confines of the last firing pattern.

The next interesting spiking behavior came at .1532 seconds in 4.10C. Here the network experienced a circular-structured multi-neuron burst within the network. The initial burst was the strongest with smaller bursts occurring after the elapsed refractory period of 3 milliseconds. After this burst of activity, the spiking activity within the network remained mostly within the bursting region. This observation led us to believe that not only did the excitation increase in certain neurons during this event, but the inhibition also increased in other neurons.

At .3632 seconds there was a different circular-structured multi-neuron burst of excitation that changed the regions in which we saw spiking activity (see 4.10D). This long burst maintained its strength through two refractory periods; occurring with the same strength at .3663 and .3694 seconds. Then, the bursting began to diminish throughout the next four refractory periods. Just like the previous case, the major spiking activity within the network confined itself within the recent bursting region due to increased inhibition in surrounding neurons.

The final interesting spiking behavior occurred at .4340 seconds. This bursting pattern faded away quickly but affected the location of spiking activity for the remainder of the simulation. Only small clusters of neurons within this region (see figure 4.10E) continued to fire.

We visually identified the interesting behavior throughout the simulation but in order to understand why this behavior may be occurring we needed to calculate the SVD of the data. Figure 4.11 plots the eigenvalues of the SVD. The eigenvalues that

caused the most variance in the network are around the first fifteen eigenvalues. Each of these eigenvalues has corresponding spatial and temporal eigenvectors. The four

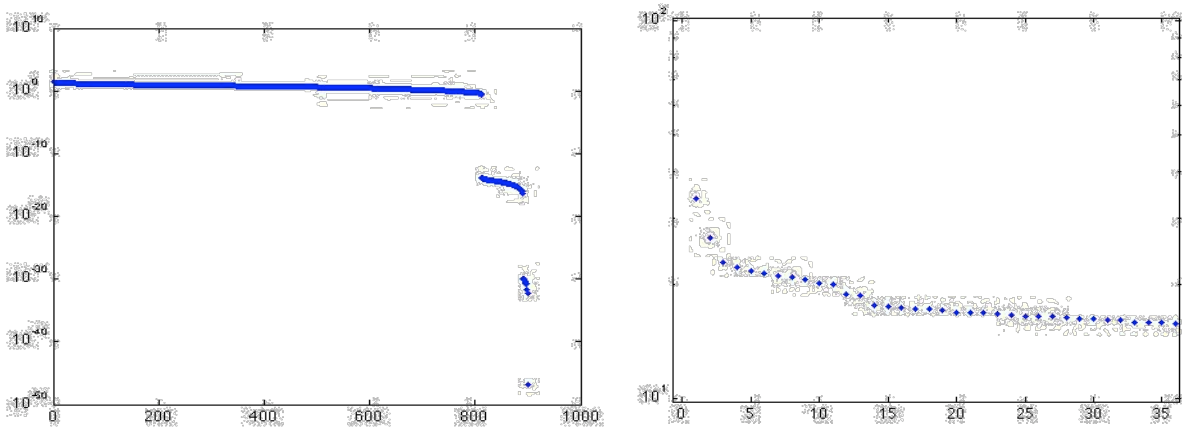


FIGURE 4.11 ONE POPULATION MODEL: EIGENVALUES OF VARIANCE A) Plot of Eigenvalues from SVD. B) A closer look of the plot of eigenvalues. Here we see which eigenvalues contribute to the most variance in the system.

general types of behavior seen in the temporal eigenvectors from the first fifteen eigenvalues are shown in Figures 4.12 – 4.16.

Figure 4.12 revealed the details behind the bursting activity in the visual simulation. Part A plots the first spatial eigenvector and we can see which neurons contributed to the most variance in the network. We clearly see the patterns from this eigenvector matching the bursting events from the simulation. In 4.12B we plot the first temporal eigenvector. It shows small oscillations of activity leading to three larger events. The largest event happened just before .4 seconds which correlates to the event seen in Figure 4.10D. We believe this event was similar to an ictal event seen in experimental results. The firing rate frequency for this eigenvector (see Fig. 4.12C) is low on average with a high frequency spike during the ictal event.

The third eigenvector's (see Fig. 4.13) spiking behavior was onset by the stimulated firing. The spiking activity maintained a steady oscillation throughout the

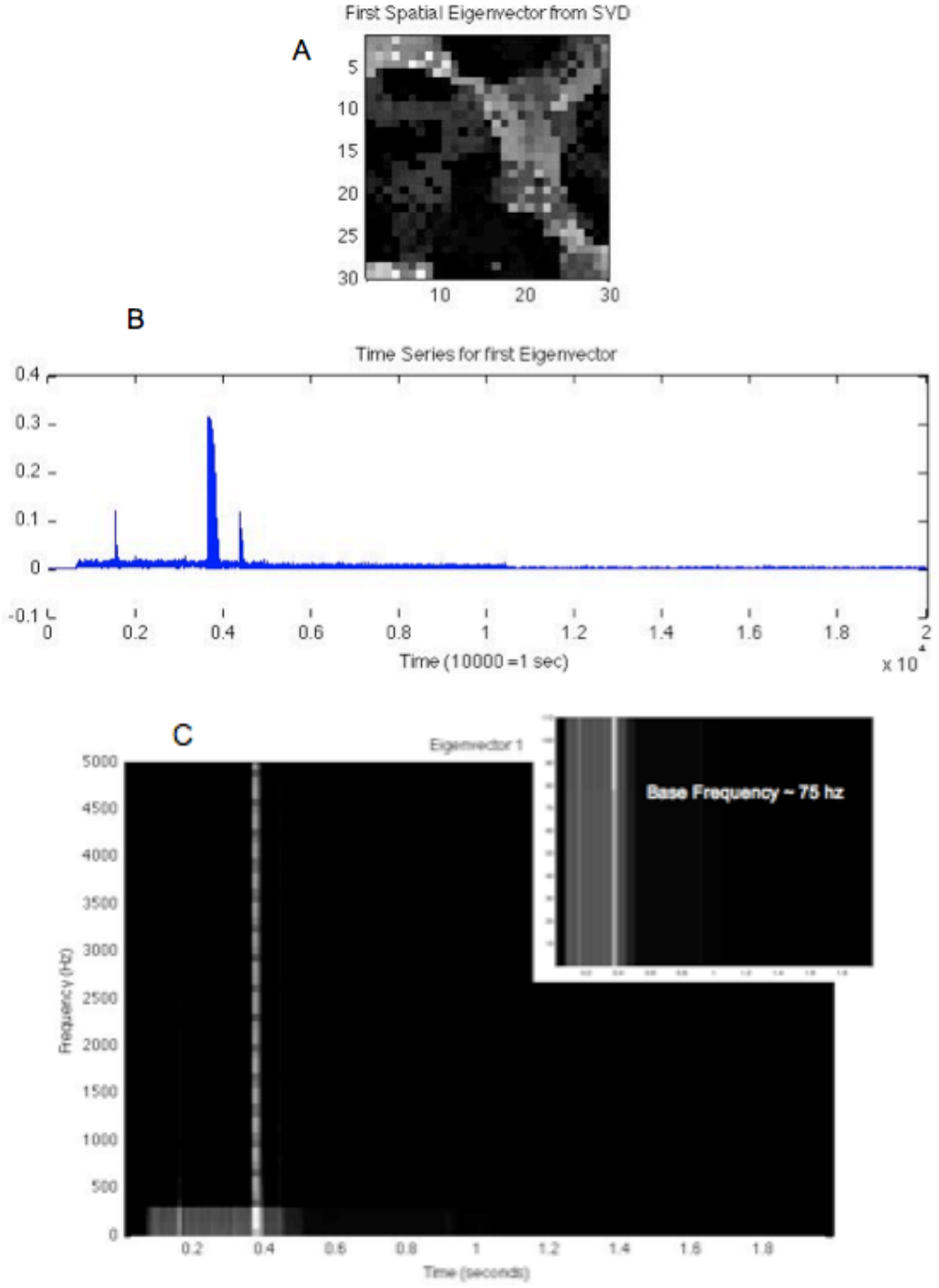


FIGURE 4.12 ONE POPULATION MODEL: FIRST EIGENVECTOR ANALYSIS. A) Shows first spatial eigenvector for firing rates B) first temporal eigenvector for firing rates C) firing rate frequency for first eigenvector

simulation. There is one spike seen in the time series data (Fig 4.13B) correlating to the first bursting event seen in Figure 4.10C. This eigenvector describes the small cluster of neurons that continue to spike throughout the length of the simulation. The base firing rate frequency is still small (around 70-80 hertz), but there is more high frequency activity. The intensity of low frequency firing rates (Fig 4.13 C) diminished after the ictal

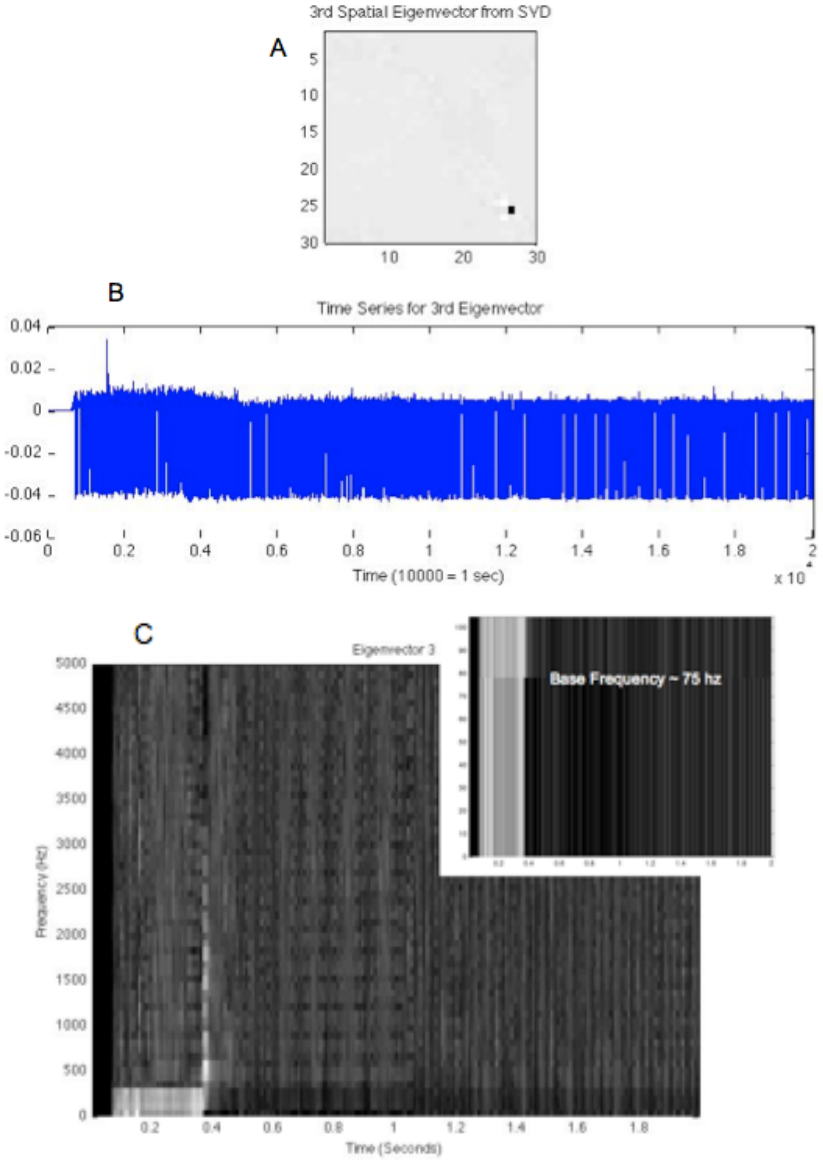


FIGURE 4.13 ONE POPULATION MODEL: THIRD EIGENVECTOR ANALYSIS. A) Shows third spatial eigenvector for firing rates B) third temporal eigenvector for firing rates C) firing rate frequency for third eigenvector

event around .4 seconds in the simulation.

The fourth eigenvector revealed there were also periods of high oscillation over time (see Figure 4.14) that weakened near the bursting events. As the first bursting event occurred, the activity died out and then began again after the event was over.

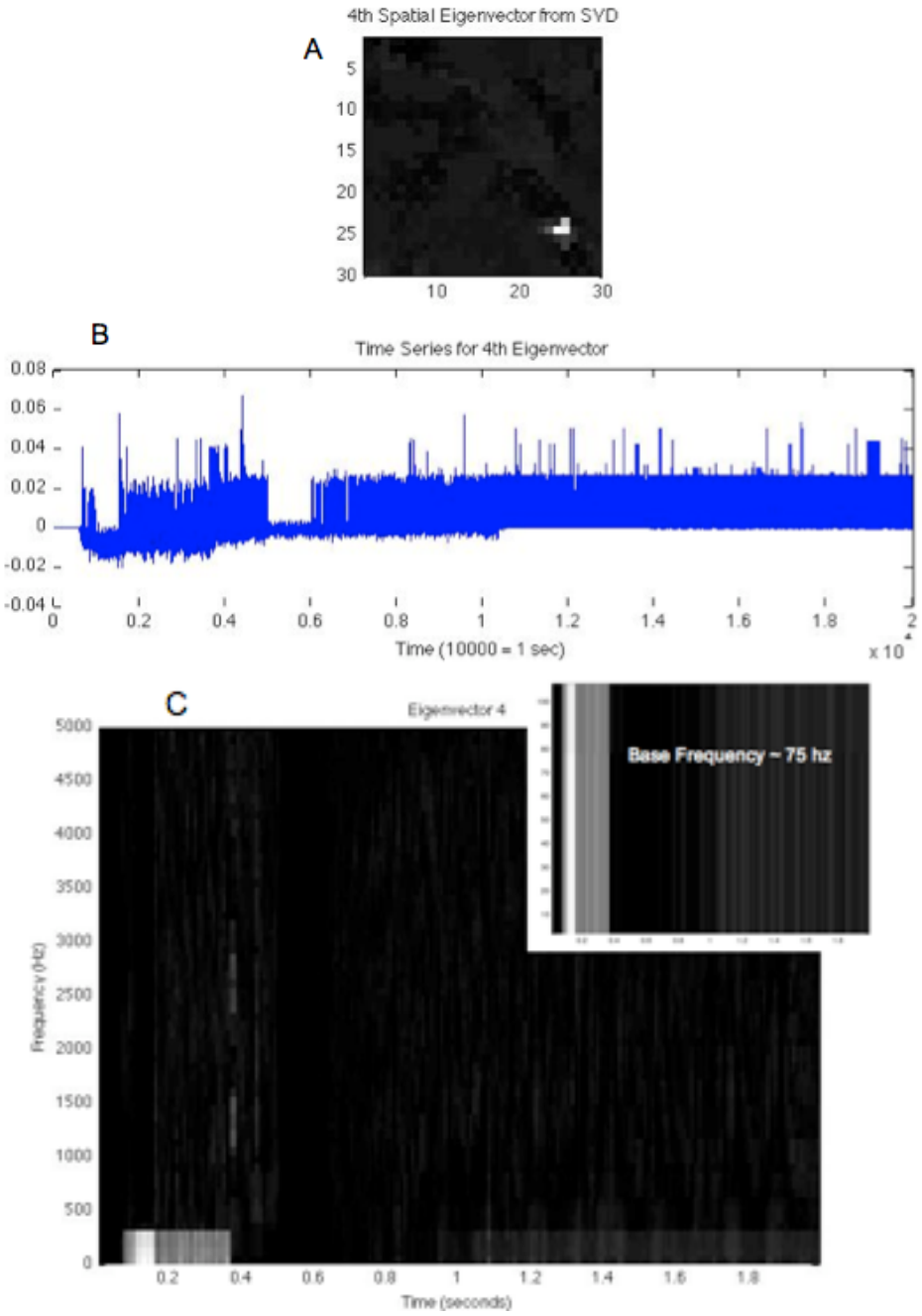
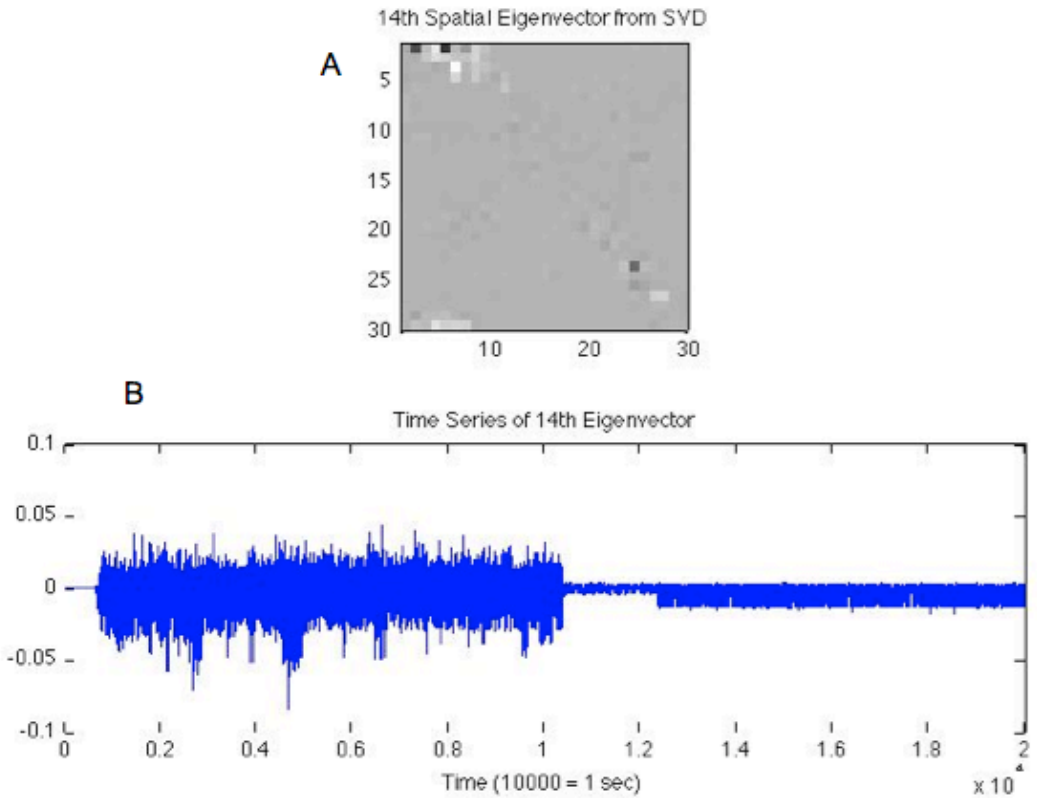


FIGURE 4.14 ONE POPULATION MODEL: THIRD EIGENVECTOR ANALYSIS. A) Shows third spatial eigenvector for firing rates B) third temporal eigenvector for firing rates C) firing rate frequency for third eigenvector

Once the third bursting event (see Fig 4.10E) was finished, the stronger oscillations weakened for approximately .1 seconds. Then the oscillations returned and continued through the length of the simulation. The low frequency firing rates (around 75 hertz) were strongest at the beginning of the simulation and faded away for a long period of time after .4 seconds. The low frequency firing rates strengthened again at approximately 1.1 seconds within the simulation.

The SVD for the fourteenth eigenvector showed that once the external input was turned off at .9 seconds, the firing rates slowed down. This confirmed what was seen during the simulation where most of the network stopped firing by the end of the simulation. Looking at the times series plot of the eigenvector (Fig. 4.15 B) we saw spiking activity until just after the one second mark. The firing rate frequency (4.15 C)



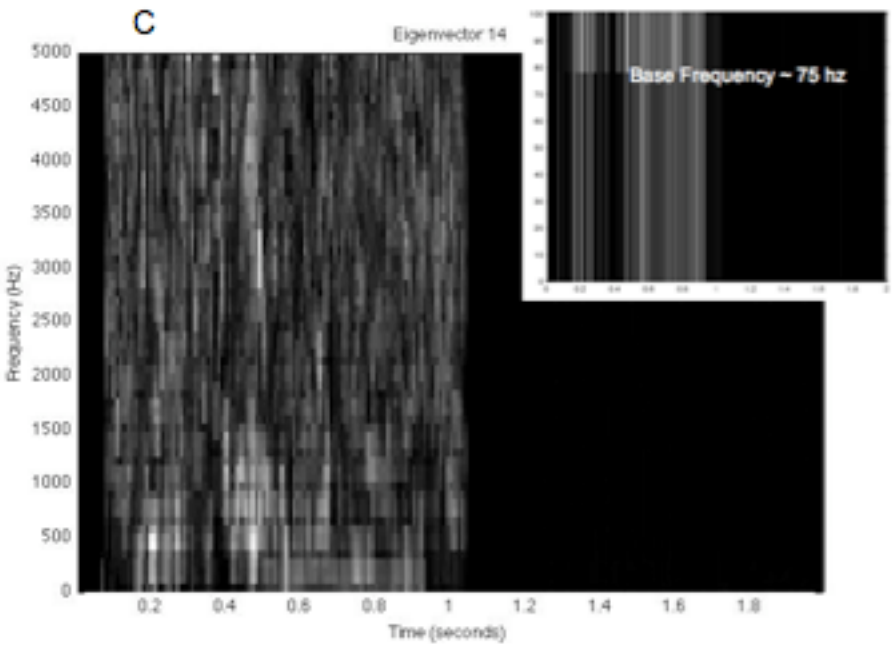


FIGURE 4.15 ONE POPULATION MODEL: FOURTEENTH EIGENVECTOR ANALYSIS. A) Shows third spatial eigenvector for firing rates B) third temporal eigenvector for firing rates C) firing rate frequency for third eigenvector

showed low to high frequencies occurring during the high amplitude activity seen in figure 4.15B.

After looking at the firing rate data, we needed to analyze the changes in calcium concentrations. We expected the changes in calcium to increase when neuronal firing increased. By examining the SVD data of the same eigenvectors seen in Figures 4.12 – 4.15 we were able to identify whether or not we were accurately predicting calcium concentrations. We discovered that the pattern of neurons within the network with the highest variability in calcium matched the pattern of neurons with the highest variability in firing rates (see Figure 4.12 and Figure 4.16).

The first eigenvector (4.16A, B) showed the increases in calcium throughout the simulation. The amount of calcium slowed to a steady state over the length of the

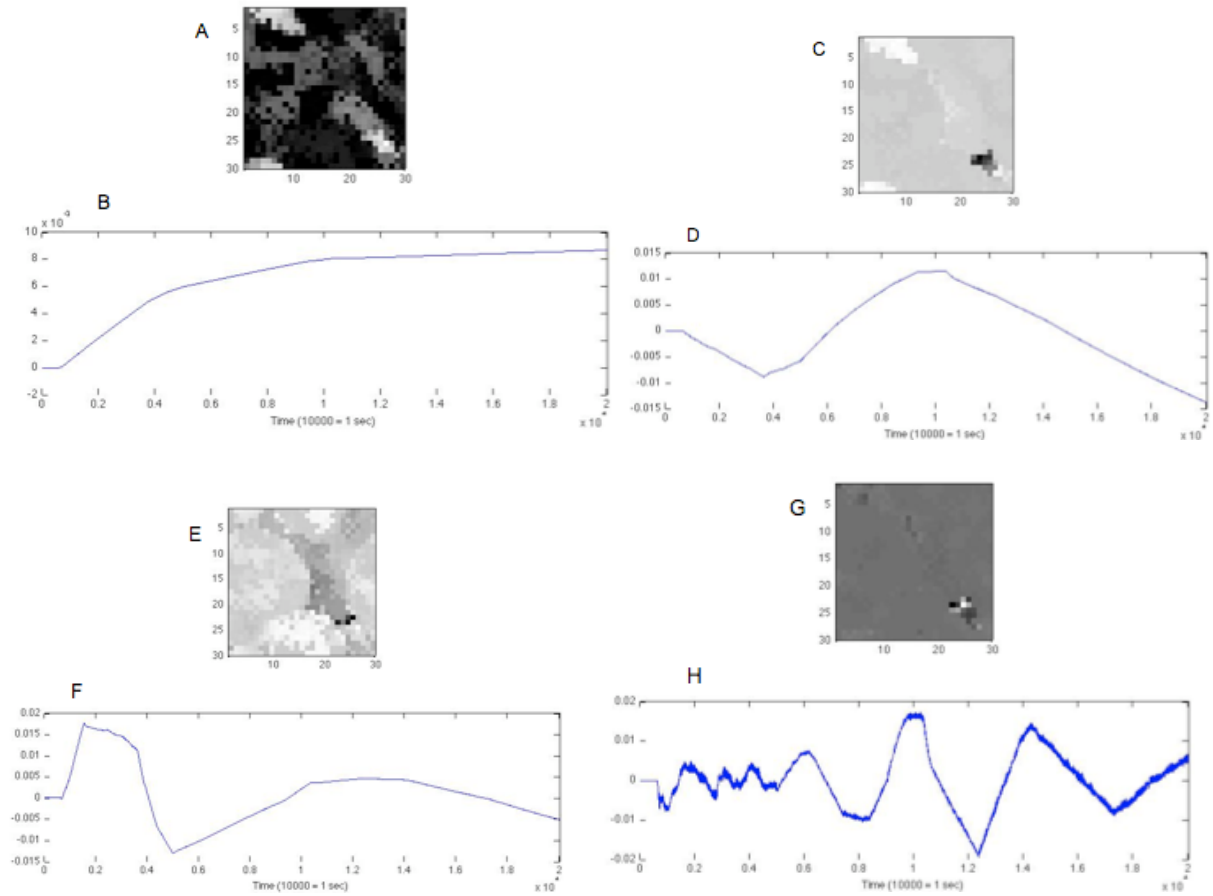


FIGURE 4.16 ONE POPULATION MODEL: EIGENVECTORS OF CALCIUM LEVELS. A) Shows first spatial eigenvector for calcium B) first temporal eigenvector for calcium C) Shows third spatial eigenvector for calcium D) third temporal eigenvector for calcium E) Shows fourth spatial eigenvector for calcium F) fourth temporal eigenvector for calcium G) Shows fourteenth spatial eigenvector for calcium H) fourteenth temporal eigenvector for calcium

simulation. This makes sense since only the one small population continued to consistently fire. The three remaining eigenvectors (3rd, 4th, and 17th in 4.16 C-H) demonstrated events during the simulation where parts of the population experienced increases or decreases in calcium concentration.

The final set of data we investigated was the YFP/CFP ratios. Based on experimental results, we expected the YFP/CFP ratios for the network to be between forty and sixty percent. Figure 4.17 shows the maximum ratio of YFP to CFP at each time step throughout the simulation. Around .8 seconds within the simulation we reached the projected range and stayed within the forty to sixty percent range through

the rest of the simulation. The graph also reveals how the YFP/CFP ratio was affected by the external input being turned off at .9 seconds.

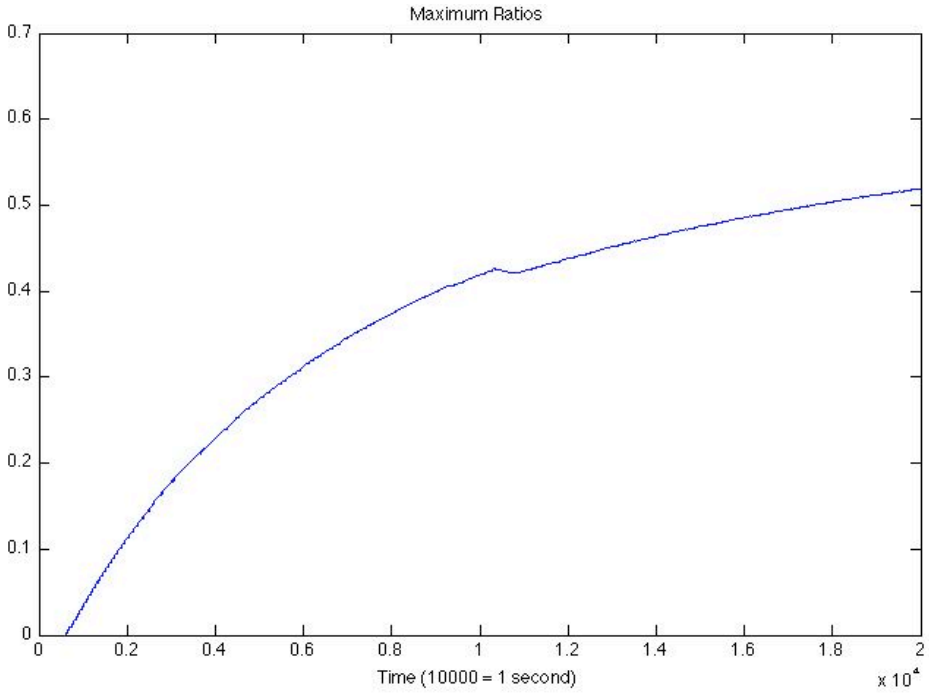


FIGURE 4.17 ONE POPULATION MODEL: MAXIMUM YFP/CFP RATIOS. Maximum ratios are in expected range of 40 to 60 percent by .8 seconds through the end of the 2-second simulation.

The results of the one population model produced propagating waves, long bursting activity, and reasonable firing rate frequencies, which were the goals of this simulation. We found a lower threshold of activity and explored how an increase in excitation affected the behavior of the network. In the following chapter, we will discuss conclusions and improvements to the model to more accurately represent the network activity within the brain of a Zebrafish.

CHAPTER 5

CONCLUSIONS

The purpose of this thesis was to create a simulation based on a mathematical model to recreate behavior seen in imaging research on seizures in the brain of a Zebrafish. The results from the previous chapter suggest that it was possible to obtain low frequency firing rates while calcium moved from one part of the population of neurons to the next. The calcium levels and emission ratios were inline with experimental data as well. However, there are improvements that can be made to the model to better represent seizure behavior in the Zebrafish.

In our model we used two different ways to represent synaptic connections within the brain. Each of these representations produced different results. Recent research has made it possible to predict the blueprint of synaptic connections within the Zebrafish brain. Therefore, an improvement to the model would be to create a connection matrix that followed this blueprint. By using a more accurate model for synaptic connections, we expect to see more precise results from the simulation.

For some parameters, the spiking activity for the neuronal network quickly died out or slowly died out throughout the length of the simulation. However, when there is a lot of excitation in the system results showed that the spiking activity of neurons could sustain itself indefinitely. This presents a problem with the model since neurons can only fire in relation to how much energy is available for action potentials. Once the

energy is depleted, the neuron is no longer able to fire. So an improvement to the model would be to keep track of this energy (known as Adenosine triphosphate (ATP)). Once ATP has been depleted, the neurons can no longer fire until more energy is stored. This would allow the simulated seizure to stop at a similar time period as an actual seizure.

The model in this thesis was a stepping-stone toward an accurate simulated model for seizures in the brain of a Zebrafish. With the improvements presented, this mathematical model will be more accurate and should produce more reasonable results in comparison to experimental data.

REFERENCES

[1] Klein, Stephen B., and Michael B. Thorne (2007) *Biological Psychology*. New York, NY: Worth Publishers.

[2] Levitan, Irwin B., and Leonard K. Kaczmarek (2002) *The neuron: cell and molecular biology*. New York, NY: Oxford University Press, Inc.

[3] Gerstner, Wulfram and Werner M. Kistler (2002) *Spiking neuron models: single neurons, populations, plasticity*. Cambridge, UK: Cambridge University Press.

[4] Lytton, William W. (2002) *From computer to brain: foundations of computational neuroscience*. New York, NY: Springer-Verlag New York, Inc.

[5] Keener, James and James Sneyd (1998) *Mathematical Physiology*. New York, NY: Springer- Verlag New York, Inc.

[6] Sundnes, Joakim (2006) *Computing the electrical activity in the heart*. The Netherlands: Springer-Verlag Berlin Heidelberg

[7] Sze, S. M. and Kwok Kwok Ng (2007) *Physics of semiconductor devices*. Hoboken, NJ: John Wiley & Sons, Inc.

[8] Koehler, Kenneth R. (2008) “College Physics for Students of Biology and Chemistry – Circuit Analysis.” Web. 9 March 2010. http://www.rwc.uc.edu/koehler/biophys.2ed/kirchhoff.html

[9] Koch C. (1999) *Biophysics of Computation: Information processing in single neurons*. Oxford University Press, NY.

[10] Shelley, Michael J. and Louis Tao. (2001) Efficient and Accurate Time-Stepping Schemes for Integrate-and-Fire Neuronal Networks. *Journal of Computational Neuroscience 11,* 111-119.

[11] Williams, Dave. University of Washington (2007). “Synaptic Transmission.” Web. 30 May 2010. http://staff.washington.edu/cdave/pbio505/Week2Presentation.xml

[12] Scientific American (1979). *The Brain*. San Francisco, CA: W.H. Freeman and Company.

[13] Exploratorium. (2010) “Neurons: A fish-eye view of the brain.” Web. 31 May 2010. www.exploratorium.edu/imaging_station

[14] Miyawaki, Griesbeck, Heim, and Tsien. (1999) Dynamic and quantitative Ca^{2+} measurements using improved cameleons. *PNAS Vol.96*, 2135-2140.

[15] Miyawaki, Llopis, Heim, McCaffery, Adams, Ikura, and Tsien. (1997) Fluorescent indicators for Ca^{2+} based on green fluorescent proteins and calmodulin. *Letters to Nature vol. 388,* 882 – 887.

[16] Brasselet, Peterman, Miyawaki, and Moerner. (2000) Single-Molecule Fluorescence Resonant Energy Transfer in Calcium Concentration Dependent Cameleon. *Journal of Physical Chemistry B vol 104,* 3676 – 3682.

[17] Yaksi, Emre and Rainer W Friedrich. (2006) Reconstruction of firing rate changes across neuronal populations by temporally deconvolved Ca^{2+} imaging. *Nature Method vol 3 no.5*, 377-383.

[18] Cheney and Kincaid. (2008) *Numerical Mathematics and Computing: Sixth Edition*. Pacific Grove, CA: Thompson Brooks/Cole.

[19] Ursino, Mauro and Giuseppe-Emiliano La Cara. (2006) Travelling waves and EEG patterns during epileptic seizure: Analysis with an integrate-and-fire neural network. *Journal of Theoretical Biology vol 242,* 171-187.

[20] Tao, Shelley, McLaughlin and Shapley. (2004) An egalitarian network model for the emergence of simple and complex cells in visual cortex. *PNAS vol 101, no. 1,* 366-371.

APPENDIX

A1. MATLAB CODE – CONNECTION MATRIX FROM 3.1

```
%Creates connectivity Matrix
function [k]=connectivity(d,e,c1,c2,c3,c4,randneuron)
k=zeros(d,d);
 for u=1:d/e
     r1=randperm(d/e);
     itself_1=r1==u;
     r1_new=r1;
     r1_new(itself_1)=[];
     rr1=r1_new(1:10);

     r2=randperm(d/e);
     del_1=randneuron(1);
     del_2=randneuron(2);
     del_3=randneuron(3);
     del_4=randneuron(4);
     indx_1=r2==del_1 | r2==del_2 | r2==del_3 | r2==del_4;
     r2_new=r2;
     r2_new(indx_1)=[];
     rr2= r2_new(1:3);

    rp1=d/e + randperm(d/e);
    itself_2=rp1==u+d/e;
    rp1_new=rp1;
    rp1_new(itself_2)=[];
    rrp1=rp1_new(1:10);

    rp2=d/e + randperm(d/e);
    indx_2=rp2==del_1|rp2==del_2 | rp2==del_3 | rp2==del_4;
    rp2_new=rp2;
    rp2_new(indx_2)=[];
    rrp2=rp2_new(1:3);

    %intra-connections (within a population)
    for tt=1:10
        k(u,rr1(tt))=abs(c1 + (c1/2)*randn);
        k(d/e+u,rrp1(tt))=abs(c3 +(c3/2)*randn);
    end

    %inter-connections (between different populations)
    for vv=1:3
```

```
        k(u,rrp2(vv))=abs(c2 +(c2/2)*randn);
        k(d/e+u,rr2(vv))=abs(c4 +(c4/2)*randn);
    end
 end
```

A2. MATLAB CODE – CONNECTION MATRIX FROM 3.2

```
%Here we create the connection matrix for excitatory connections
%*based off of Ursino paper
function [Ex]=ConnectSpaceExcite(d,e,w_ex)
%**We are assuming e is even

format long
A=zeros([e e e e]);
for i=1:e
  for j=1:e
    k=rand;
    p=i-(e/2);
    q=j-(e/2);
    for l=p:e+p-1
     for m=q:e+q-1
         dist=sqrt((i-l)^2+(j-m)^2);
         if 0~=mod(l,e)
           if 0==mod(m,e)
     A(i,j,mod(l,e),e)=w_ex*(.5+k)*exp(-(1/4)*dist^2);
           else
   A(i,j,mod(l,e),mod(m,e))=w_ex*(.5+k)*exp(-(1/4)*dist^2);
           end
         else
           if 0==mod(m,e)
            A(i,j,e,e)=w_ex*(.5+k)*exp(-(1/4)*dist^2);
           else
          A(i,j,e,mod(m,e))=w_ex*(.5+k)*exp(-(1/4)*dist^2);
           end
        end
      end
    end
  end
end
Ex=reshape(A,[d d]);
for ii=1:d
    Ex(ii,ii)=0;
end

end
```

```
%Here we created the connection matrix for inhibitory
connections
function [In]=ConnectSpaceInhibit(d,e,w_in)
%**We are assuming e is even

format long;
A=zeros([e e e e]);
for i=1:e
 for j=1:e
    k=rand;
    p=i-(e/2);
    q=j-(e/2);
   for l=p:e+p-1
     for m=q:e+q-1
         dist=sqrt((i-l)^2+(j-m)^2);
        if 0~=mod(l,e)
         if 0==mod(m,e)
         A(i,j,mod(l,e),e)=w_in*(.5+k)*exp(-(1/16)*dist^2);
         else
  A(i,j,mod(l,e),mod(m,e))=w_in*(.5+k)*exp(-(1/16)*dist^2);
         end
       else
         if 0==mod(m,e)
            A(i,j,e,e)=w_in*(.5+k)*exp(-(1/16)*dist^2);
         else
         A(i,j,e,mod(m,e))=w_in*(.5+k)*exp(-(1/16)*dist^2);
         end
      end
    end
  end
end
end
In=reshape(A,[d d]);
for ii=1:d
    In(ii,ii)=0;
end
end
```

A3. MATLAB CODE – RK4

```
function [t,count_t_spike,calcium,ratios]=
PopNeurons_Final_revised(a, b, N,lim, d, e, w_ex, w_in,
external,refract)
```

```
feature accel on
format long

%h=time step, t=time, vthres= voltage threshold for spiking,
vrest= voltage reset value tau is time constant for excitation,
tau_inhibit for inhibition

h=(b-a)/N;
t=a:h:b;
vthres=1;
vrest=0;
tau=.001;
tau_inhibt=.002;

%v=voltage, u=estimate of voltage used in resetting after a
spike, g,w = 2 dimensional conductance system of equations
%g_e=conductance from excitation, g_i=conductance from
inhibition

v_old=zeros(d, 1);
v=zeros(d,1);
u_old=zeros(d,1);
u=zeros(d, 1);

g_old=zeros(d,1);
g=zeros(d,1);
w_old=zeros(d,1);
w=zeros(d,1);
g_e=zeros(d,1);

g_inhibt_old=zeros(d,1);
g_inhibt=zeros(d,1);
w_inhibt_old=zeros(d,1);
w_inhibt=zeros(d,1);
g_i=zeros(d,1);

local_g2=zeros(d,1);
local_g_e2=zeros(d,1);
local_g_inhibt2=zeros(d,1);
local_g_i2=zeros(d,1);
local_g3=zeros(d,1);
local_g_e3=zeros(d,1);
local_g_inhibt3=zeros(d,1);
local_g_i3=zeros(d,1);

%Variables for rk4 for finding voltage of each neuron
k1=zeros(d,1);
k2=zeros(d,1);
```

```
k3=zeros(d,1);
k4=zeros(d,1);

%Variables for rk4 for finding new voltage after a spike for
each neuron
j1=zeros(d,1);
j2=zeros(d,1);
j3=zeros(d,1);
j4=zeros(d,1);

%Calcium
cal=zeros(d,1);
calcium=zeros(d,N);

%Variables for rk4 to find calcium for next time step
y1=zeros(d,1);
y2=zeros(d,1);
y3=zeros(d,1);
y4=zeros(d,1);

%dummy substitution variables
s1=zeros(d,1);
s2=zeros(d,1);
s3=zeros(d,1);
s4=zeros(d,1);
s5=zeros(d,1);
s6=zeros(d,1);
s7=zeros(d,1);
s8=zeros(d,1);
s9=zeros(d,1);
s10=zeros(d,1);
s11=zeros(d,1);
s12=zeros(d,1);

%stores the ratios
ratios=zeros(d,N);
r_min=0;
r_max=1;

%g_ext= External conductance into system; 3x3 matrix of neurons
in center
g_ext=zeros(d,1);
%Trying to determine the middle to give external conductance to
a 3x3
%matrix of neurons
Marray=1:d;
Matrix=reshape(Marray,e,e)';
mid= size(Matrix,1)/2;
```

```
pre_mid=mid-1;
post_mid=mid+1;
neu=Matrix(pre_mid:post_mid,pre_mid:post_mid);

%Counter for spikes for each neuron. Firing rates are calculated
over each time step as the number of spikes per time step
t_spike=zeros(d,N);
count_t_spike=zeros(d,N);

%Finds the connectivity matrix Ex for excitatory and In for
inhibitory connections
Ex=ConnectSpaceExcite(d,e,w_ex);
In=ConnectSpaceInhibit(d,e,w_in);

for i=1:N %Outermost for-loop for time
    %noise for the system right now we are having no noise
    noise=zeros(d,1);

    %noise=poissrnd(.1,d);

%if/else statement to cut on/off the external conductance
    if i <= lim
        for p=1:9
         g_ext(neu(p))=external;
        end

    else
        for p=1:9
         g_ext(neu(p))=0;
        end
    end

    %now we calculate the conductance and the voltage at the
beginning of the time step for each neuron influenced by the
external current and synaptic connections. The inhibitory
connections have added noise in the calculation and are not
connected to an external current. It uses rk4 to find the
voltage for the next time step.

    g_e(1:d)=g_ext(1:d)+(Ex(:,:)*g(:));
    g_i(1:d)=(In(:,:)*g_inhibt(:))+noise;

        v_old=v;

        alph=50*ones(d,1);
        alpha1=(alph+g_e+g_i);
        beta1=(14/3)*g_e-(2/3)*g_i;
        k1(1:d)=-alpha1.*v_old(1:d)+beta1;
```

```
        local_g2(1:d)=exp(-((h/2))/tau)*g+((h/2)/tau)*exp(-
((h/2))/tau)*w;
        local_g_e2(1:d)=g_ext(1:d)+Ex(:,:)*local_g2(:);
        local_g_inhibt2(1:d)=exp(-
((h/2))/tau_inhibt)*g_inhibt+(((h/2))/tau_inhibt)*exp(-
((h/2))/tau_inhibt)*w_inhibt;
        local_g_i2(1:d)=In(:,:)*local_g_inhibt2(:)+noise;
        alpha2=(alph+local_g_e2+local_g_i2);
        beta2=(14/3)*local_g_e2-(2/3)*local_g_i2;
        k2(1:d)=-alpha2.*(v_old(1:d)+k1(1:d)*(h/2))+beta2;

        k3(1:d)=-alpha2.*(v_old(1:d)+k2(1:d)*(h/2))+beta2;

        local_g3(1:d)=exp(-((h))/tau)*g+((h)/tau)*exp(-
((h))/tau)*w;
        local_g_e3(1:d)=g_ext(1:d)+Ex(:,:)*local_g3(:);
        local_g_inhibt3(1:d)=exp(-
((h))/tau_inhibt)*g_inhibt+(((h))/tau_inhibt)*exp(-
((h))/tau_inhibt)*w_inhibt;
        local_g_i3(1:d)=In(:,:)*local_g_inhibt3(:)+noise;
        alpha3=(alph+local_g_e3+local_g_i3);
        beta3=(14/3)*local_g_e3-(2/3)*local_g_i3;
        k4(1:d)=-alpha3.*(v_old(1:d)+k3(1:d)*h)+beta3;



v(1:d)=v_old(1:d)+(h/6)*(k1(1:d)+2*k2(1:d)+2*k3(1:d)+k4(1:d));


%Refractory Period so a neuron doesn't spike more than
physically possible
        if i>refract
          idx=logical(sum(t_spike(1:d,i-refract:i),2)~= 0);
          v(idx)=vrest;
        end
        if i<refract
            idx=logical(sum(t_spike(1:d,1:i),2)~=0);
            v(idx)=vrest;
        end
    %This section searches through the voltage level for each
neuron at the current time step to determine if the voltage has
passed the threshold. If it has, then we know a spike has
occurred during the time step. The code below finds the
estimated spike time (t_spike) and recalculates the voltage,
conductance, and calcium levels for that neuron.

%If a spike has not occurred, no new calculation is needed.
```

```
    %Identifies whether voltage is above threshold
    idx2=find(v(1:d)>=vthres);
    idx3=find(v(1:d)<vthres);

     if (~isempty(idx3))          %no spike
%Calculated conductance for excitatory based on original voltage
           g_old(idx3)=g(idx3);
           w_old(idx3)=w(idx3);
           g(idx3)=exp(-((h))/tau)*g_old(idx3)+((h)/tau)*exp(-
((h))/tau)*w_old(idx3);
           w(idx3)=exp(-(h)/tau).*w_old(idx3);

%Calculated conductance for inhibitory based on original voltage
           g_inhibt_old(idx3)=g_inhibt(idx3);
           w_inhibt_old(idx3)=w_inhibt(idx3);
           g_inhibt(idx3)=exp(-
((h))/tau_inhibt)*g_inhibt(idx3)+(((h))/tau_inhibt)*exp(-
((h))/tau_inhibt)*w_inhibt(idx3);
           w_inhibt(idx3)=exp(-
(h)/tau_inhibt).*w_inhibt_old(idx3);

           %Calculated calcium levels
            cal_old=cal;
            y1(idx3)=dcaldt(0,cal_old(idx3));
            y2(idx3)=dcaldt(h/2, cal_old(idx3)+y1(idx3)*h/2);
            y3(idx3)=dcaldt(h/2, cal_old(idx3)+y2(idx3)*h/2);
            y4(idx3)=dcaldt(h, cal_old(idx3)+h*y3(idx3));

cal(idx3)=cal_old(idx3)+(h/6)*(y1(idx3)+2*y2(idx3)+2*y3(idx3)+y4
(idx3));

     end

     if (~isempty(idx2))        %we have a spike!
           indx=size(idx2,1);

       for c=idx2'
          t_spike(c,i)=fzero(@(x) vthres -(v_old(c)*(1-
alpha1(c)*x-3*x^2/(h^2)+2*x^2*alpha1(c)/h+2*x^3/(h^3)-
x^3*alpha1(c)/(h^2))+v(c)*(3*x^2/(h^2)-x^2*alpha3(c)/h -
2*x^3/(h^3) -x^3*alpha3(c)/(h^2))+ beta1(c)*x - 2*x^2*beta1(c)/h
+ x^2*beta3(c)/h + x^3*beta1(c)/(h^2) +
x^3*beta3(c)/(h^2)),.0001);
       end

%Calculated conductance for excitatory connections after a spike
           g_old(idx2)=g(idx2);
```

```
            w_old(idx2)=w(idx2);
            g(idx2)=exp(-(h)/tau).*g_old(idx2)+((h)/tau).*exp(-
(h)/tau).*w_old(idx2)+((h-t_spike(idx2,i))/tau^2).*exp(-(h-
t_spike(idx2,i))/tau);
            w(idx2)=exp(-(h)/tau).*w_old(idx2)+(1/tau).*exp(-(h-
t_spike(idx2,i))/tau);

%Calculated conductance for inhibitory connections after a spike
            g_inhibt_old(idx2)=g_inhibt(idx2);
            w_inhibt_old(idx2)=w_inhibt(idx2);
            g_inhibt(idx2)=exp(-
(h)/tau_inhibt).*g_inhibt_old(idx2)+((h)/tau_inhibt).*exp(-
(h)/tau_inhibt).*w_inhibt_old(idx2)+((h-
t_spike(idx2,i))/tau_inhibt^2).*exp(-(h-
t_spike(idx2,i))/tau_inhibt);
            w_inhibt(idx2)=exp(-
(h)/tau_inhibt).*w_inhibt_old(idx2)+(1/tau_inhibt).*exp(-(h-
t_spike(idx2,i))/tau_inhibt);

 %If tspike < h/2, need to calculate alpha and beta as if a
 %spike has occurred by half way
             alph4=50*ones(indx,1);
             g_half(1:d,1)=exp(-
(h/2)/tau).*g_old+((h/2)/tau).*exp(-(h/2)/tau).*w_old+(((h/2)-
t_spike(:,i))/tau^2).*exp(-((h/2)-t_spike(:,i))/tau);
             g_e_half(1:indx,1)=g_ext(idx2)+Ex(idx2,:)*g_half(:);
             g_inhibt_half(1:d,1)=exp(-
(h/2)/tau_inhibt).*g_inhibt_old+((h/2)/tau_inhibt).*exp(-
(h/2)/tau_inhibt).*w_inhibt_old+(((h/2)-
t_spike(:,i))/tau_inhibt^2).*exp(-((h/2)-
t_spike(:,i))/tau_inhibt);

g_i_half(1:indx,1)=In(idx2,:)*g_inhibt_half(:)+noise(idx2);

alpha4=(alph4+g_e_half(1:indx,1)+g_i_half(1:indx,1));
             beta4=(14/3)*g_e_half(1:indx,1)-
(2/3)*g_i_half(1:indx,1);

%Tells us which alpha and beta to use when re-calculating
             %voltage
            if t_spike < h/2
             alpha2=alpha4;
             beta2=beta4;
            end

%Calculating alpha and beta at end of time step after a spike
             local_g_e5(1:indx,1)=g_ext(idx2)+Ex(idx2,:)*g(:);
```

```
local_g_i5(1:indx,1)=In(idx2,:)*g_inhibt(:)+noise(idx2);

alpha5=(alph4+local_g_e5(1:indx,1)+local_g_i5(1:indx,1));
            beta5=(14/3)*local_g_e5(1:indx,1)-
(2/3)*local_g_i5(1:indx,1);

%dummy variables to help me calculate voltage
           s1(idx2)= -alpha1(idx2);
           s2(idx2)=beta1(idx2);
           s3(idx2)=-alpha2(1:indx)-
(alpha2(1:indx).*s1(idx2))*h/2;
           s4(idx2)=(-alpha2(1:indx)*(h/2).*s2(idx2)) +
beta2(1:indx);
           s5(idx2)= -alpha4(1:indx)-
(alpha4(1:indx)*(h/2).*s3(idx2));
           s6(idx2)=(-alpha4(1:indx)*(h/2).*s4(idx2))
+beta4(1:indx);
           s7(idx2)= -alpha5(1:indx)-alpha5(1:indx)*h.*s5(idx2);
           s8(idx2)= -alpha5(1:indx)*h.*s6(idx2)+ beta5(1:indx);
           s9(idx2)= ones(indx,1) + (h/6)*s1(idx2) +
(h/3)*s3(idx2) + (h/3)*s5(idx2) + (h/6)*s7(idx2);
           s10(idx2)= (h/6)*s2(idx2) + (h/3)*s4(idx2) +
(h/3)*s6(idx2) + (h/6)*s8(idx2);
           s11(idx2)= -alpha3(idx2).*s9(idx2);
           s12(idx2)= -alpha3(idx2).*s10(idx2) + beta3(idx2);

           %re-calculating v_n
           u_old(idx2)=(vrest*ones(indx,1) -
s2(idx2).*t_spike(idx2,i) -
(3*s10(idx2).*(t_spike(idx2,i)).^2)*(1/h^2) +
(2*s2(idx2).*(t_spike(idx2,i)).^2)*(1/h) -
(s12(idx2).*(t_spike(idx2,i)).^2)*(1/h) +
(2*s10(idx2).*(t_spike(idx2,i)).^3)*(1/h^3) -
(s2(idx2).*(t_spike(idx2,i)).^3)*(1/h^2) -
(s12(idx2).*(t_spike(idx2,i)).^3)*(1/h^2))./(ones(indx,1) +
s1(idx2).*t_spike(idx2,i) +
(3*s9(idx2).*(t_spike(idx2,i)).^2)*(1/h^2) -
(3*(t_spike(idx2,i)).^2)*(1/h^2) -
(2*s1(idx2).*(t_spike(idx2,i)).^2)*(1/h) +
(s11(idx2).*(t_spike(idx2,i)).^2)*(1/h) -
(2*s9(idx2).*(t_spike(idx2,i)).^3)*(1/h^3) +
(2*(t_spike(idx2,i)).^3)*(1/h^3) +
(s1(idx2).*(t_spike(idx2,i)).^3)*(1/h^2) +
(s11(idx2).*(t_spike(idx2,i)).^3)*(1/h^2));

%The v below is the newly calculated voltage (v_n+1)
           j1(idx2)=-alpha1(idx2).*u_old(idx2)+ beta1(idx2);
           j2(idx2)=-
```

```
alpha2(1:indx).*(u_old(idx2)+j1(idx2)*(h/2))+ beta2(1:indx);
            j3(idx2)=-
alpha4(1:indx).*(u_old(idx2)+j2(idx2)*(h/2))+ beta4(1:indx);
            j4(idx2)=-alpha5(1:indx).*(u_old(idx2)+j3(idx2)*h)+
beta5(1:indx);

u(idx2)=u_old(idx2)+(h/6)*(j1(idx2)+2*j2(idx2)+2*j3(idx2)+j4(idx
2));
            v(idx2)=u(idx2);

%This is the newly calculated level of calcium released.
            cal_old=cal;
            y1(idx2)=dcaldt1(0,cal_old(idx2),indx);
            y2(idx2)=dcaldt1(h/2,
cal_old(idx2)+h*y1(idx2)*(1/2),indx);
            y3(idx2)=dcaldt1(h/2,
cal_old(idx2)+h*y2(idx2)*(1/2),indx);
            y4(idx2)=dcaldt1(h, cal_old(idx2)+h*y3(idx2),indx);

cal(idx2)=cal_old(idx2)+(h/6)*(y1(idx2)+2*y2(idx2)+2*y3(idx2)+y4
(idx2));

    end

            %finds firing rate for each time step
           for mm=1:d
            if t_spike(mm,i)~=0
            count_t_spike(mm,i)=1;
            end
           end

%holds onto calcium levels
    calcium(:,i)=cal(1:d);

% finds ratios
   ratios(:,i) = (cal(1:d)*r_max + 10^(-
6.5)*ones(d,1)*r_min)./(10^(-6.5)*ones(d,1)+ cal(1:d));
end

end

%Calcium diff equations. When a spike has not occurred, dcaldt
is used. When a spike has occurred, dcaldt1 is used to evaluate
calcium.
function dcaldt=dcaldt(t,cal)
%if t==0
dcaldt=-cal*(1/2);
%else
```

```
%end

end
function dcaldt1=dcaldt1(t,cal,indx)
delta=.00001*ones(indx,1);
dcaldt1=-cal*(1/2) + delta;
end
```